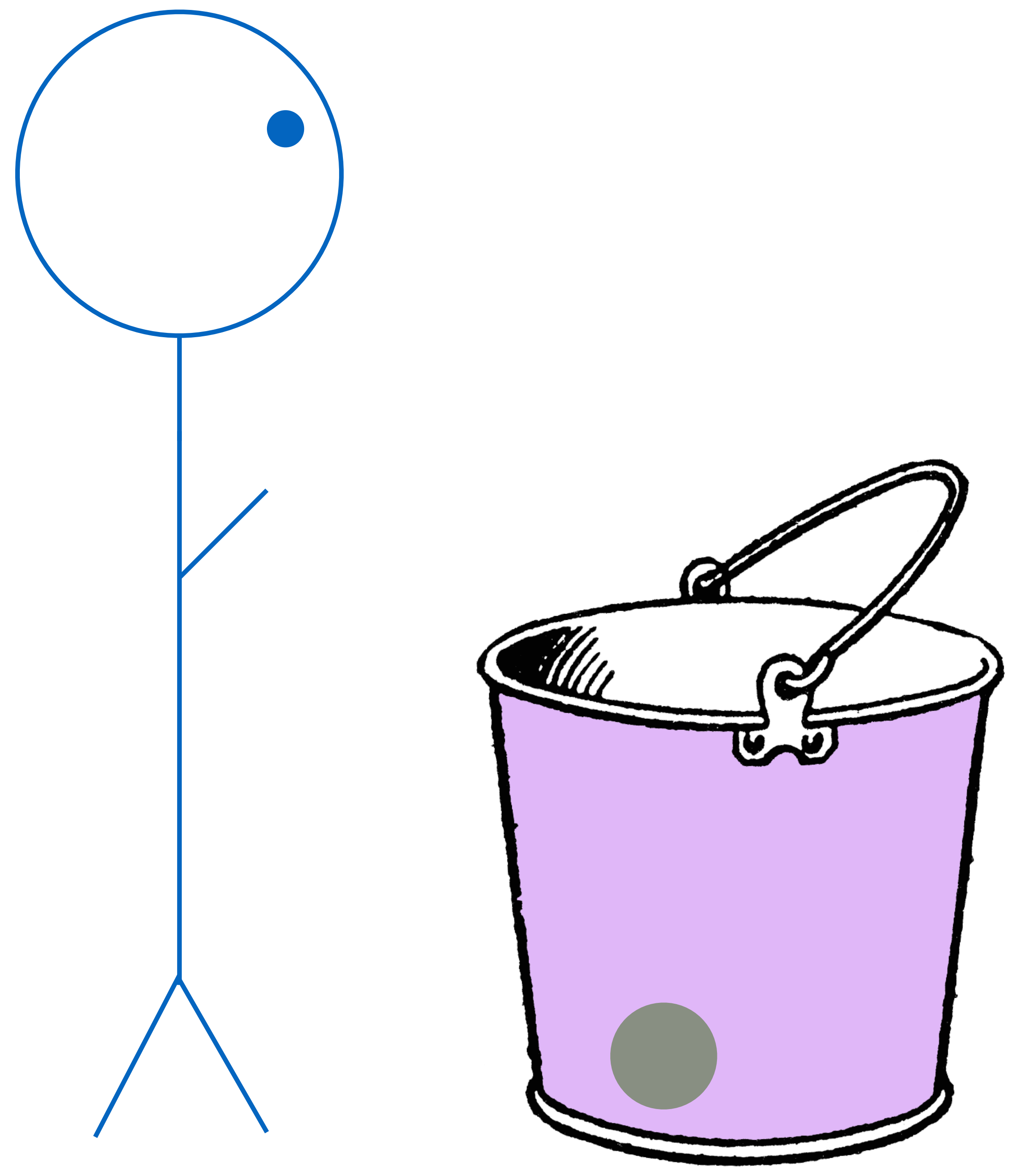
Semaphores

Ryan Eberhardt
July 26, 2021

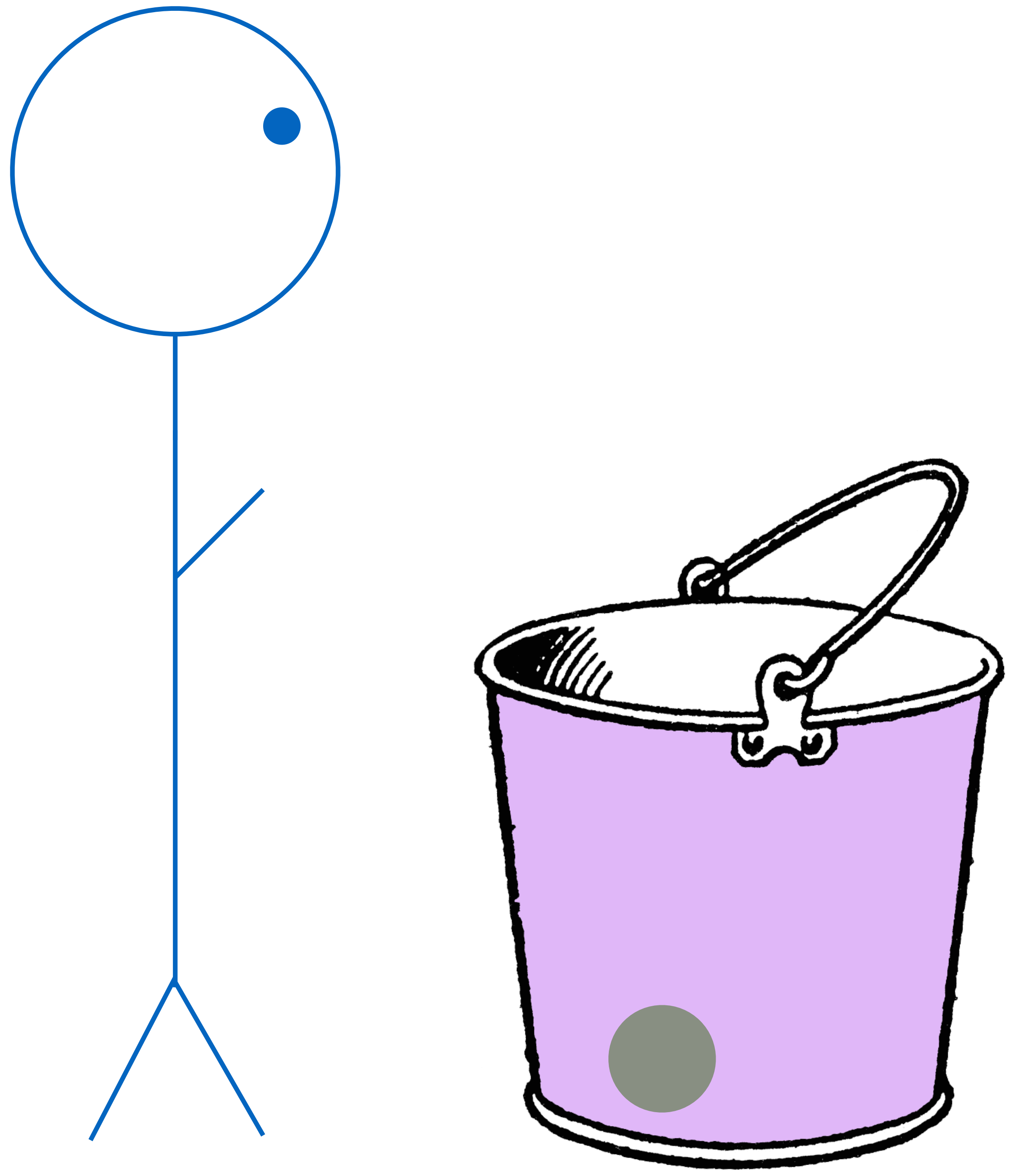
Semaphores


thread1

Semaphores

semaphore.wait()

If necessary, waits for a ball to be added to the bucket; then, takes the ball

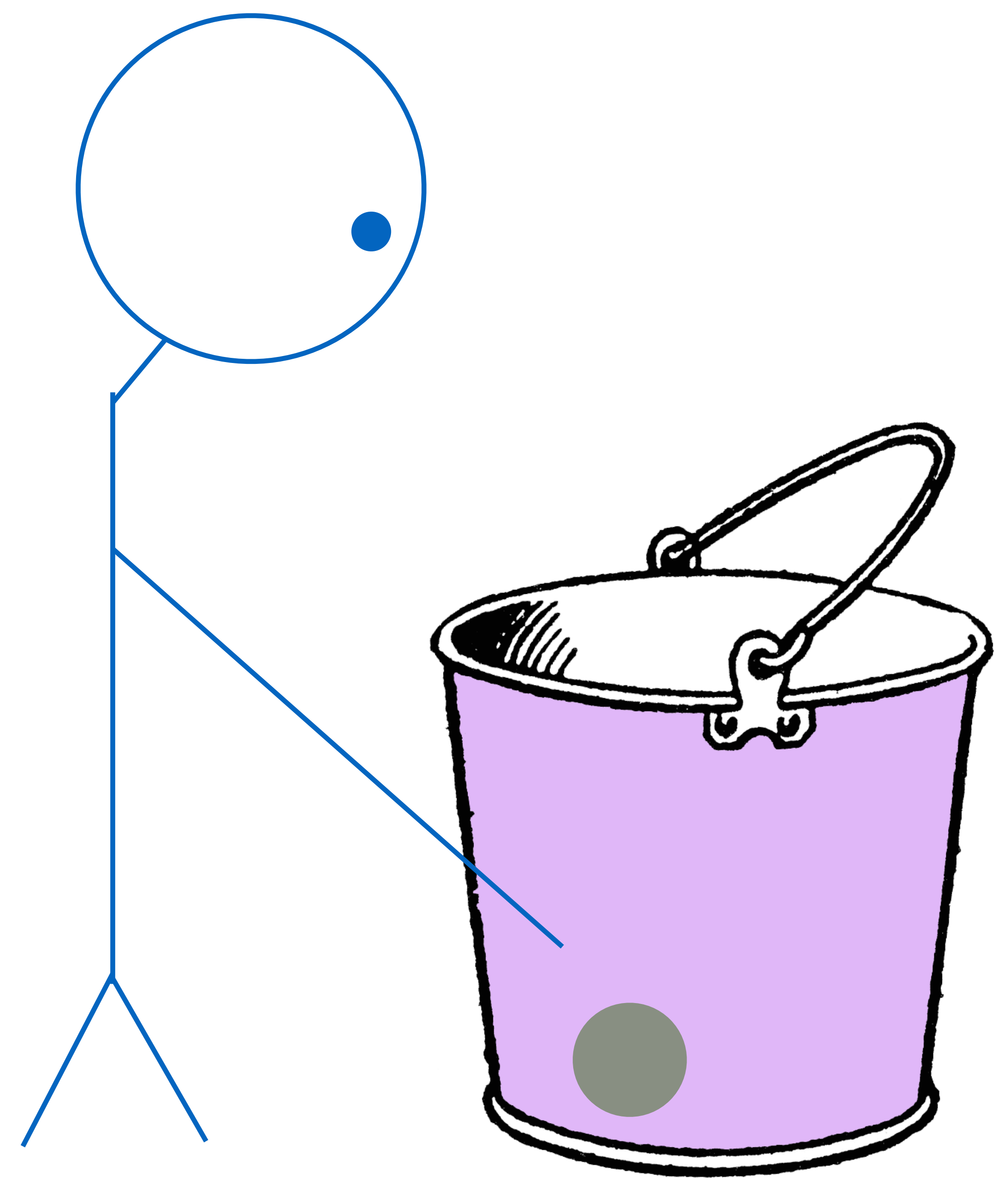


thread1

Semaphores

semaphore.wait()

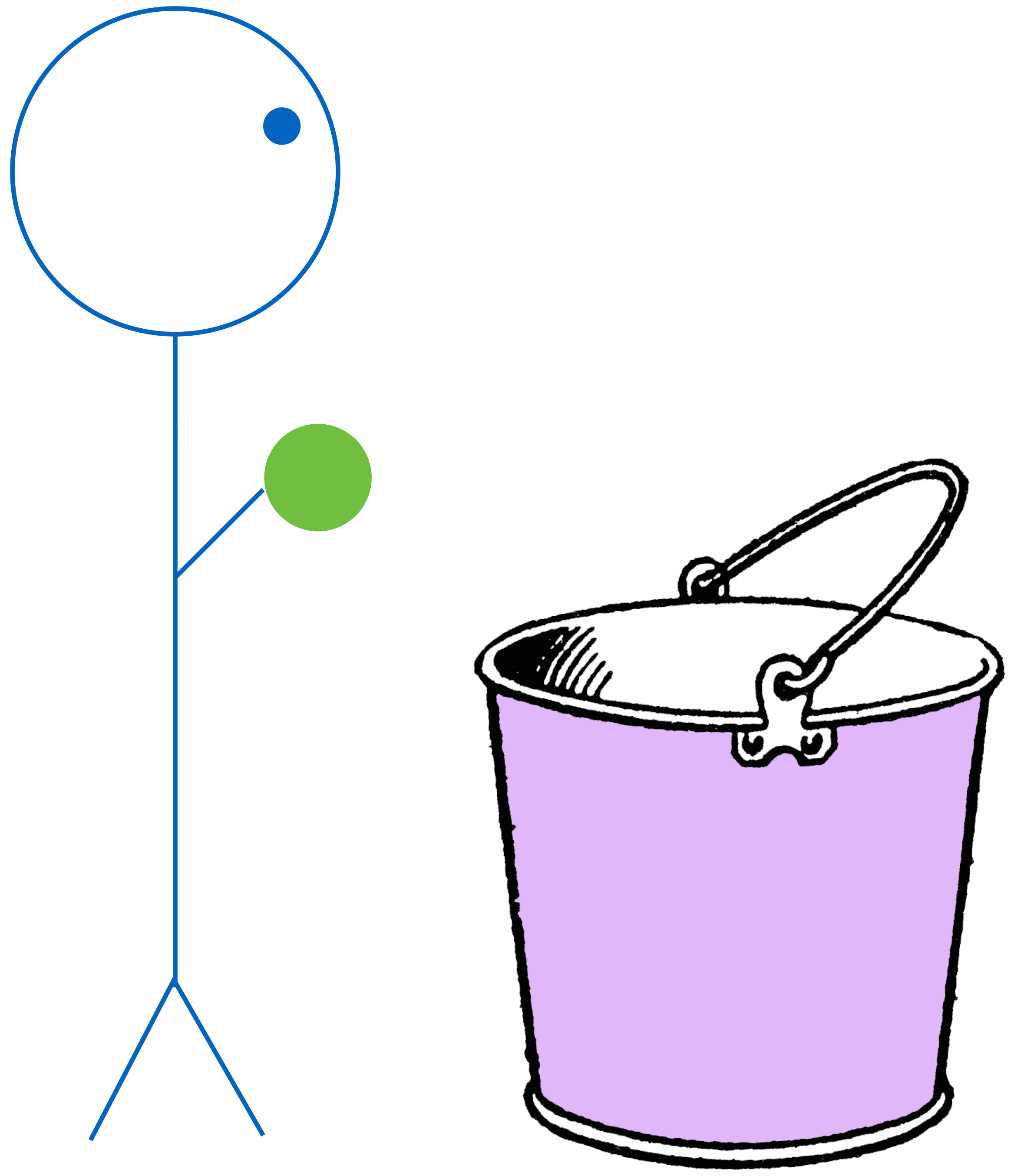
If necessary, waits for
a ball to be added to
the bucket; then,
takes the ball



Semaphores

semaphore.wait()

If necessary, waits for a ball to be added to the bucket; then, takes the ball

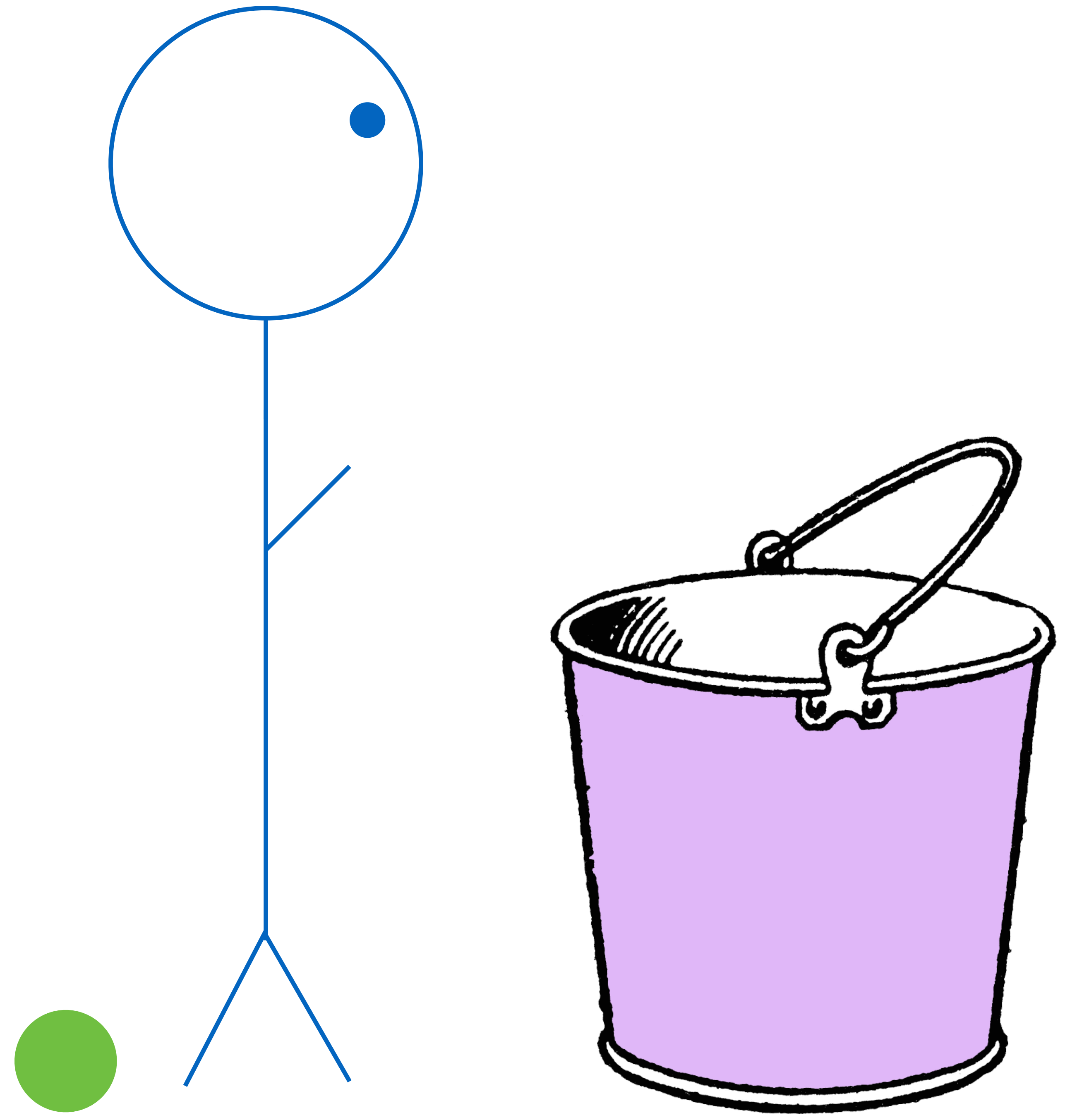


thread1

Semaphores

semaphore.wait() (again)

If necessary, waits for a ball to be added to the bucket; then, takes the ball

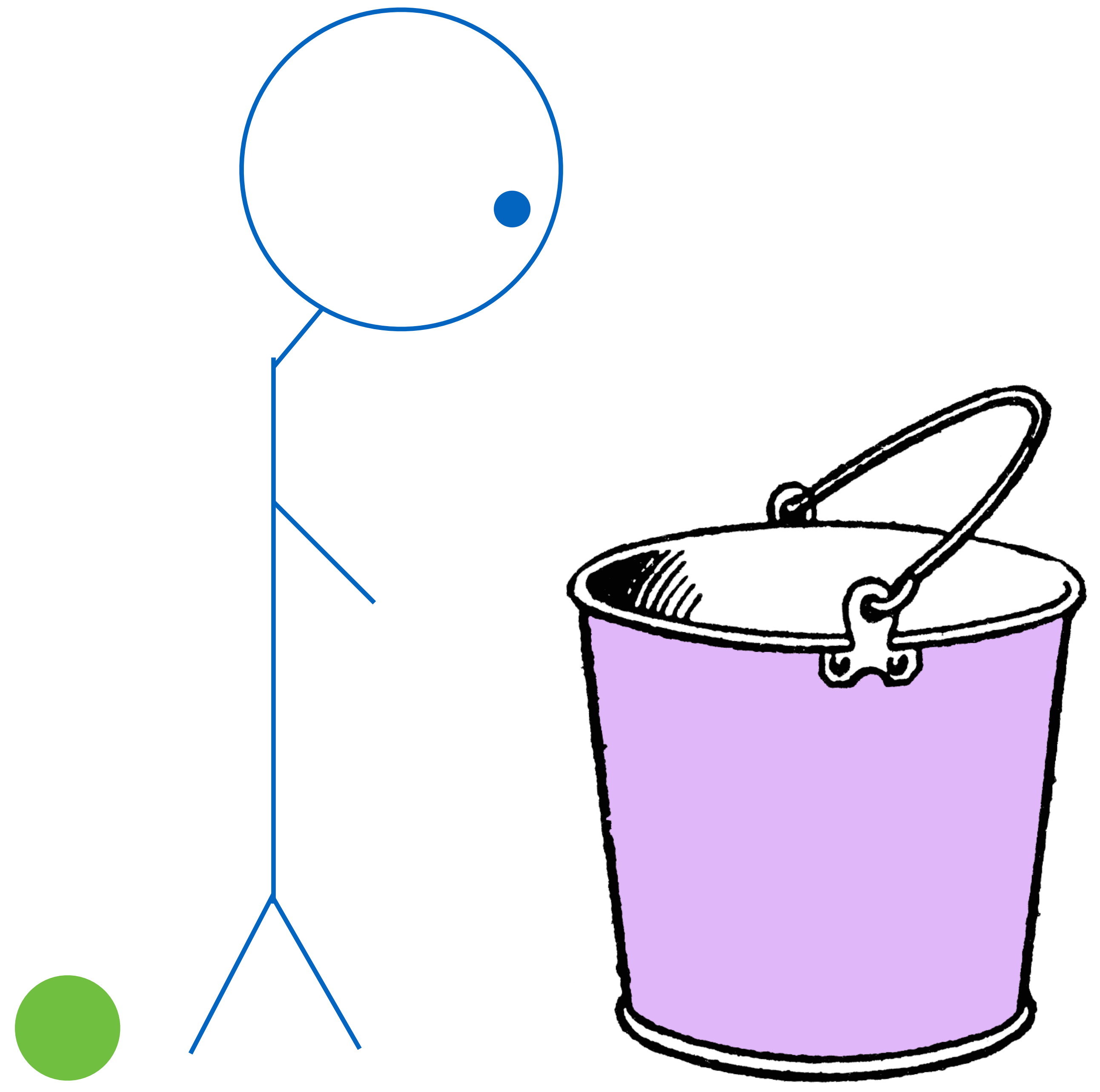


thread1

Semaphores

semaphore.wait() (again)

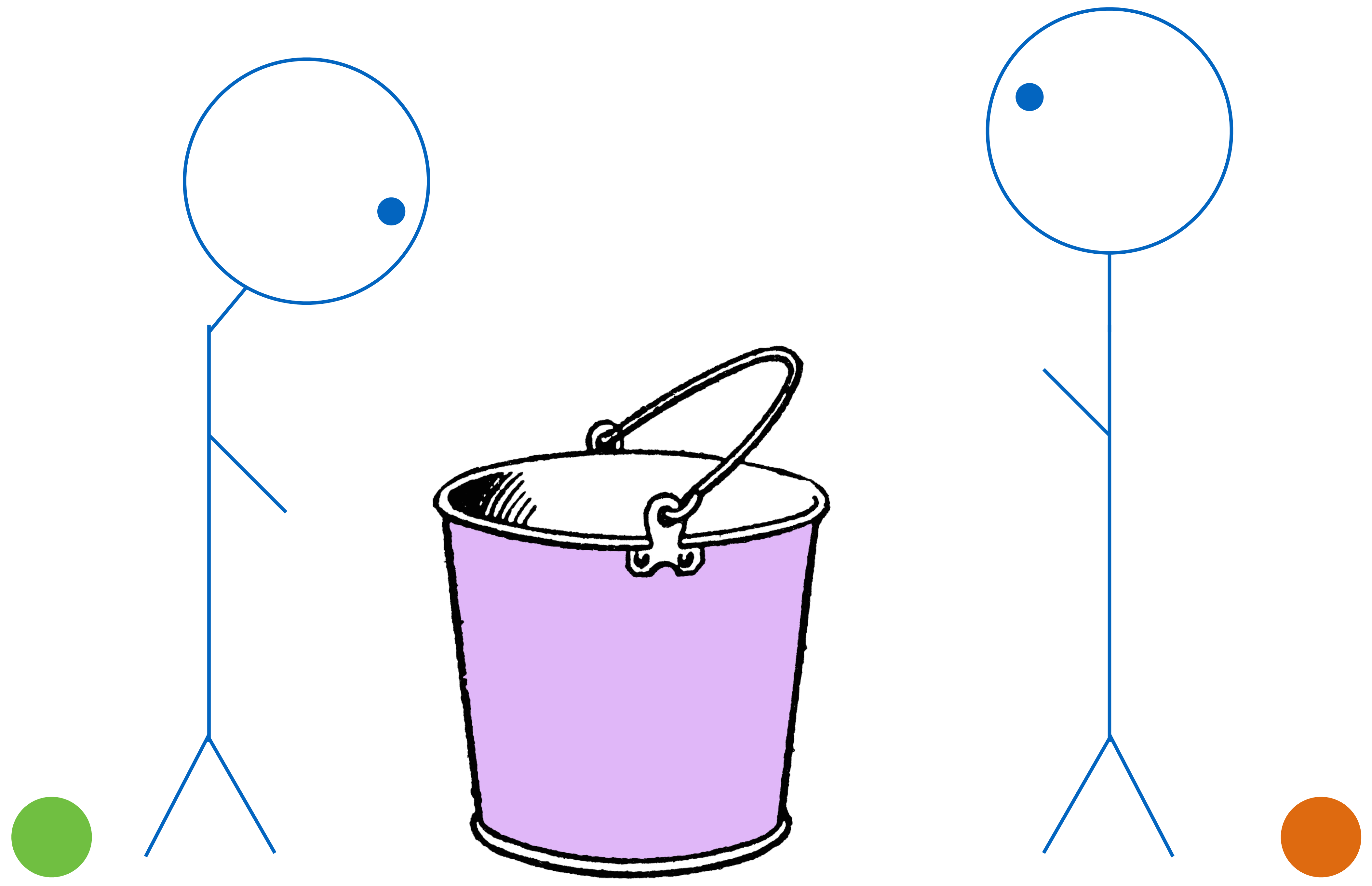
If necessary, waits for a ball to be added to the bucket; then, takes the ball



thread1 (blocked)

Semaphores

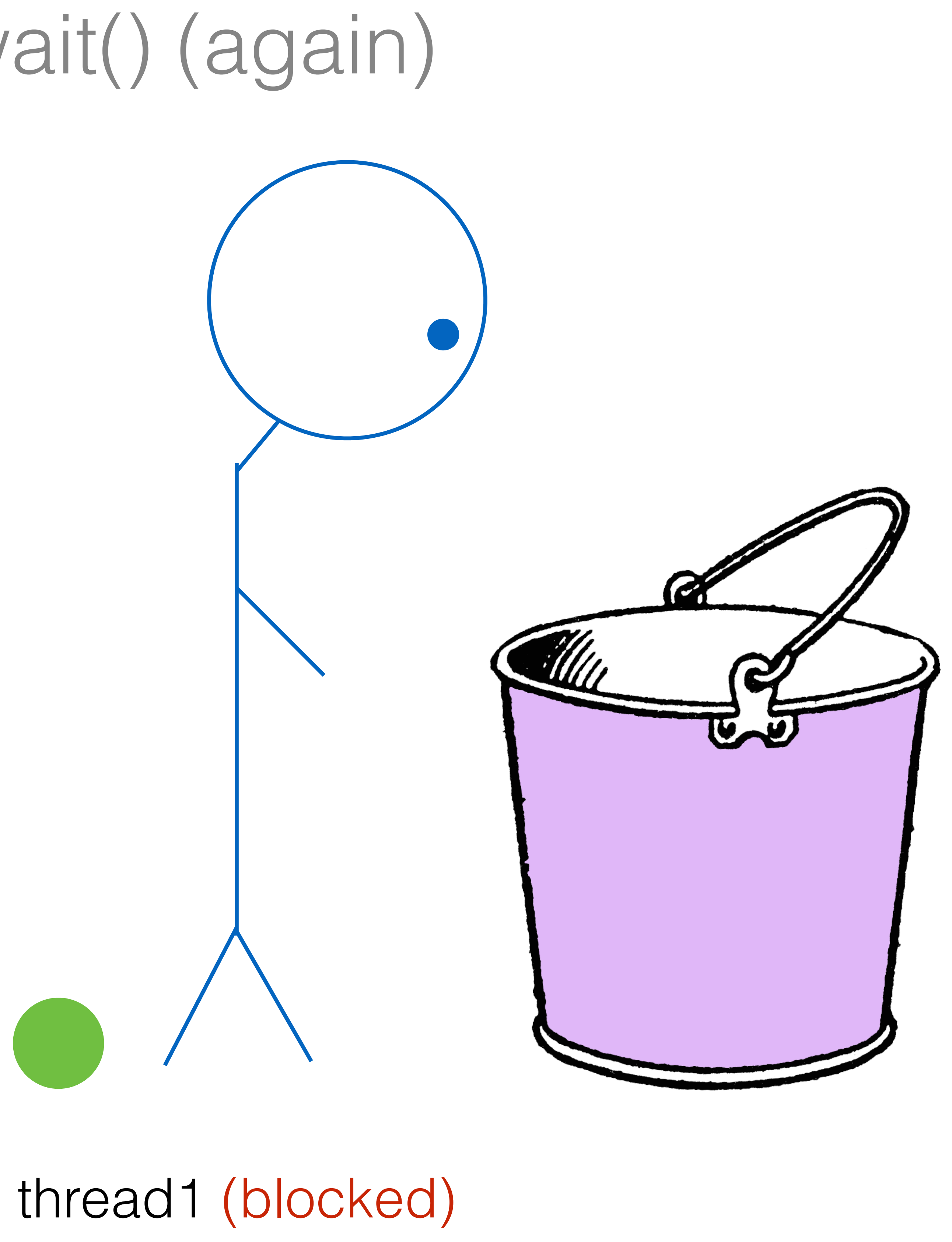
semaphore.wait() (again)

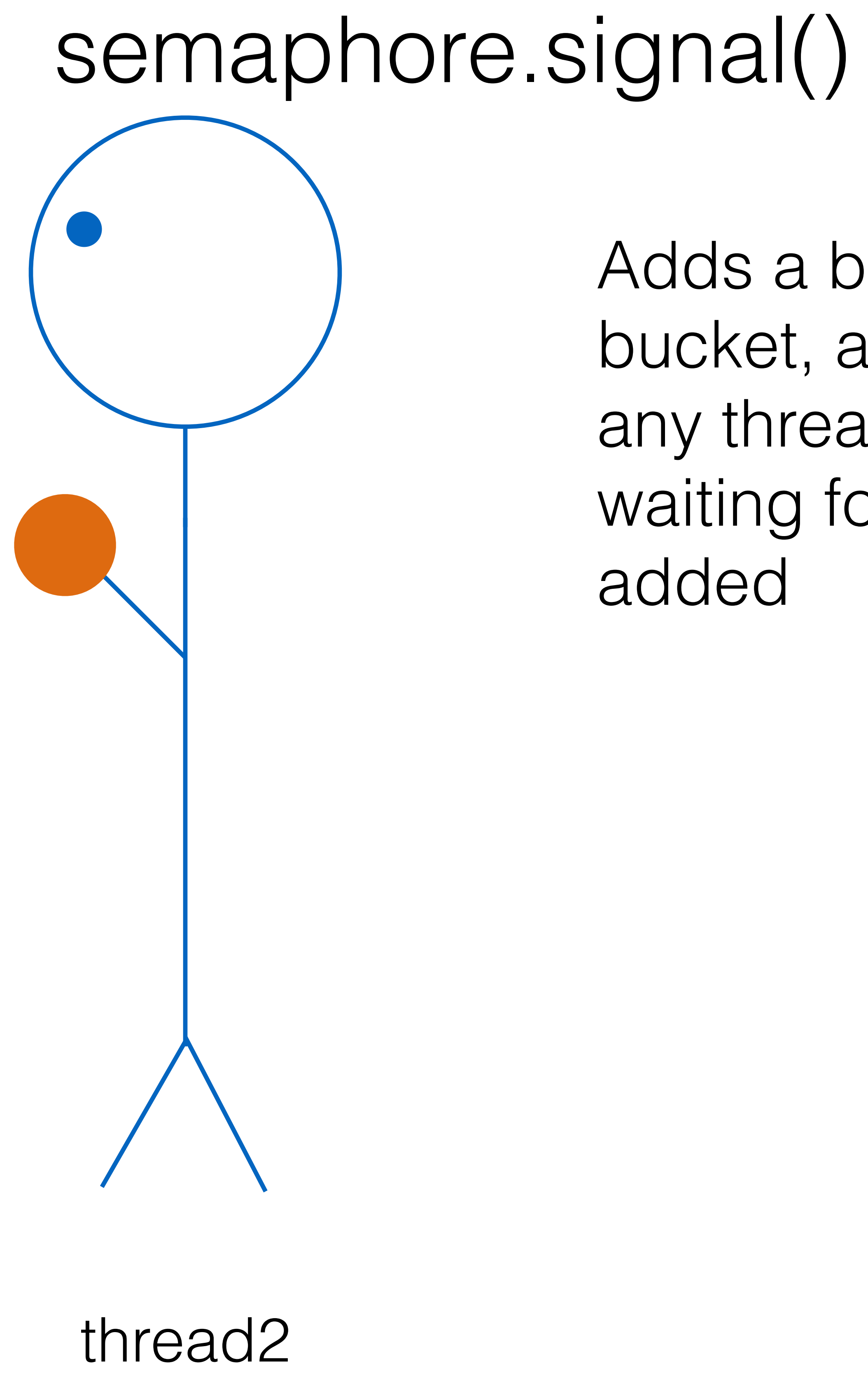


thread1 (blocked)

thread2

Semaphores



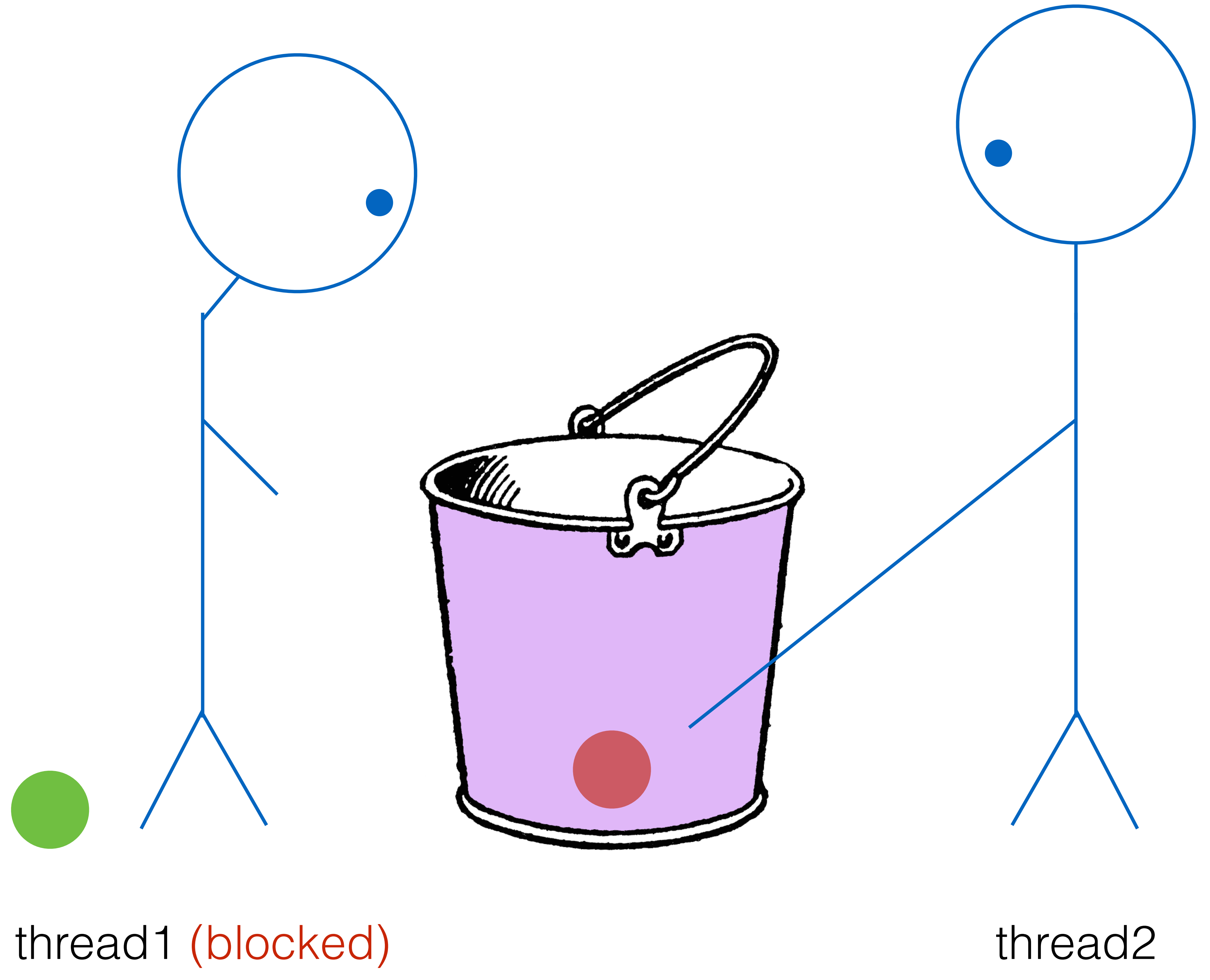


Adds a ball to the bucket, and wakes up any threads that were waiting for one to be added

Semaphores

semaphore.wait() (again)

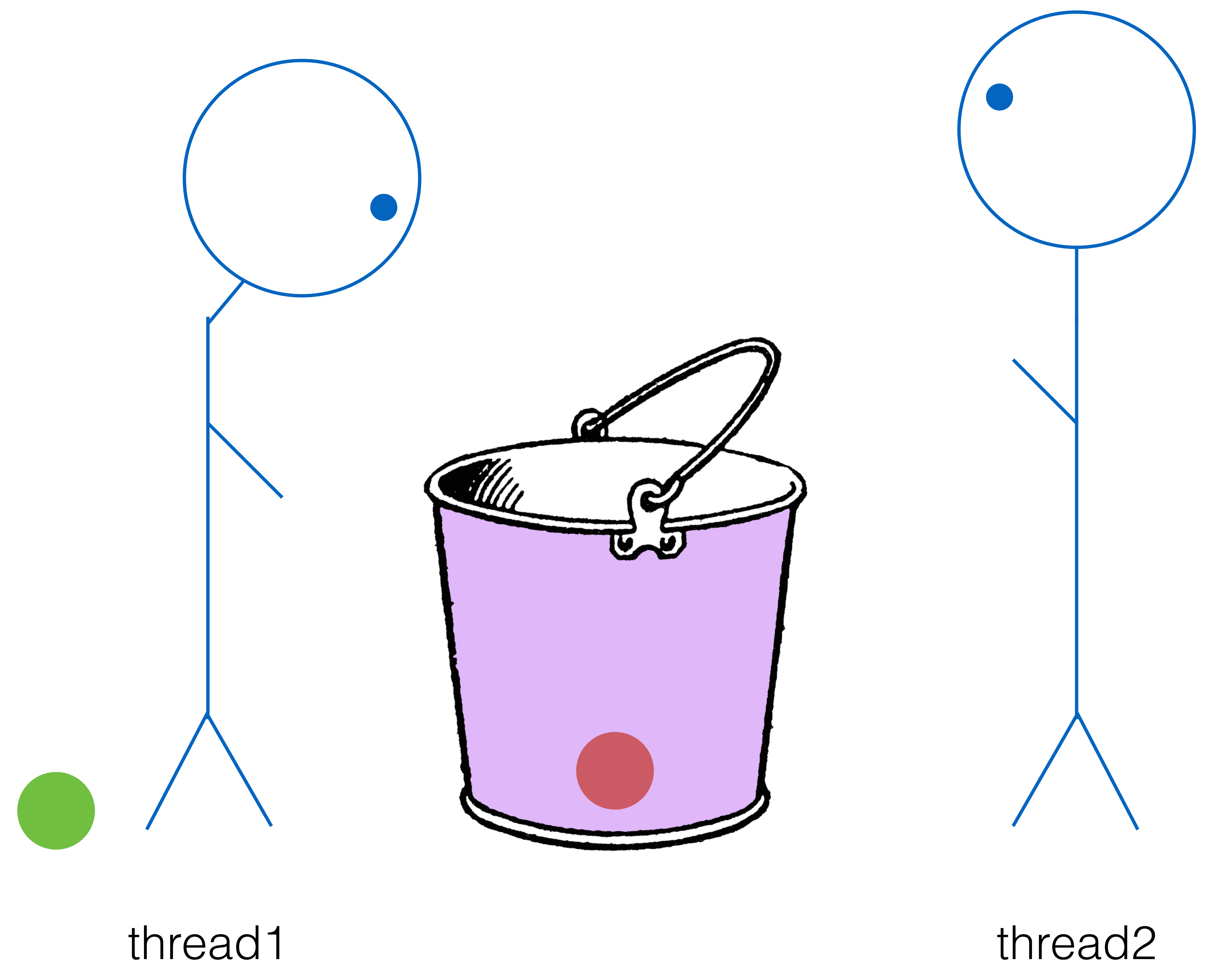
semaphore.signal()



Adds a ball to the bucket, and wakes up any threads that were waiting for one to be added

Semaphores

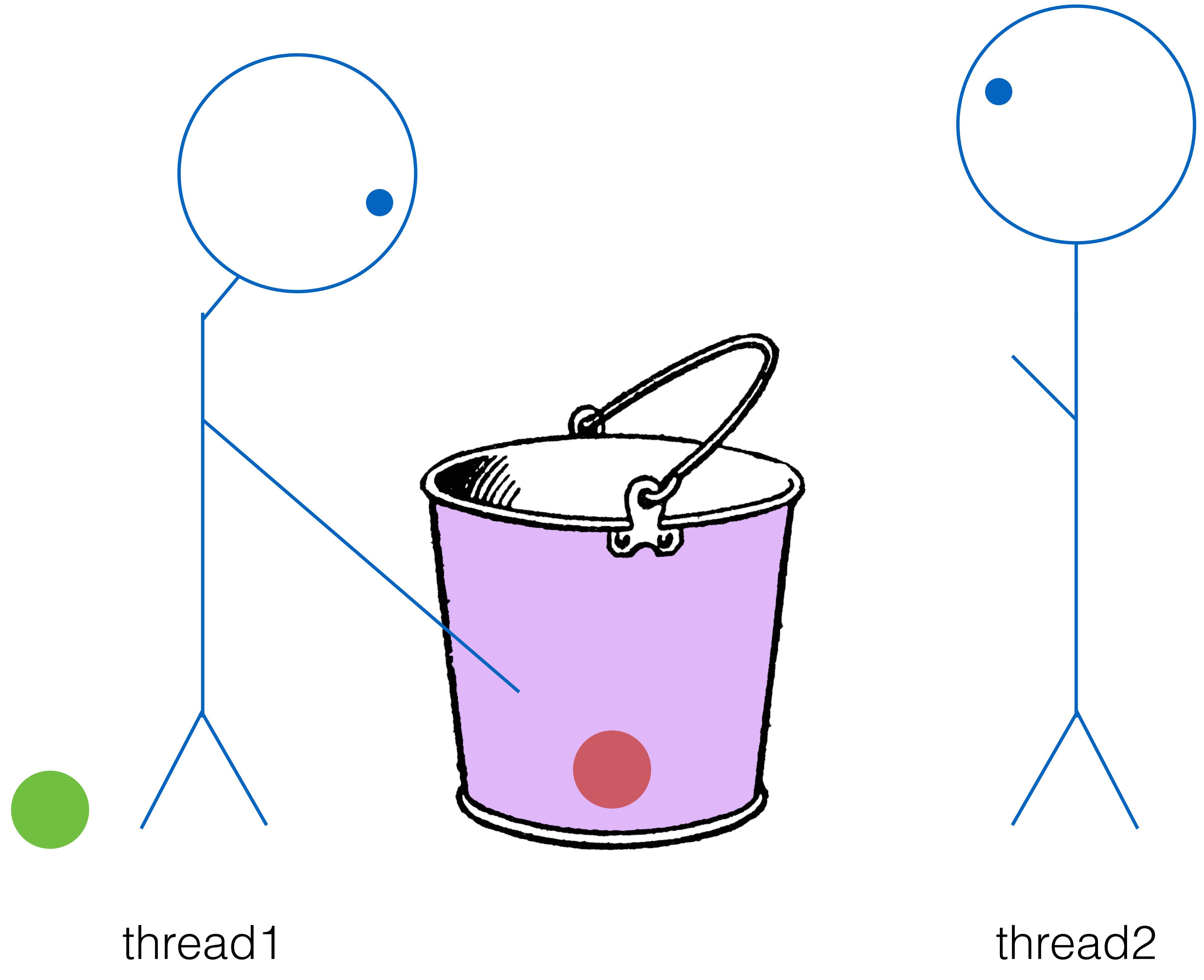
semaphore.wait() (again)



Adds a ball to the bucket, and wakes up any threads that were waiting for one to be added

Semaphores

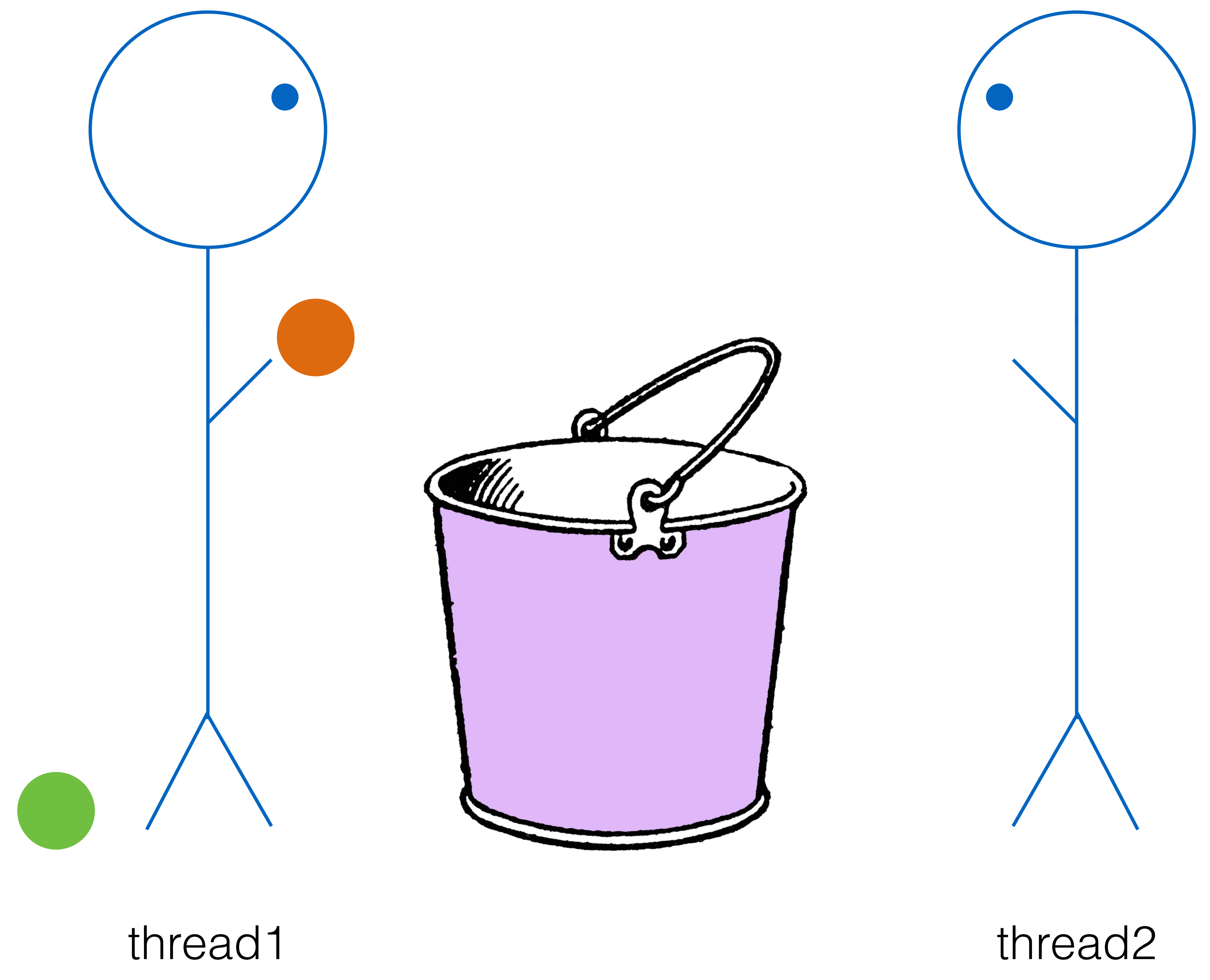
semaphore.wait() (again)



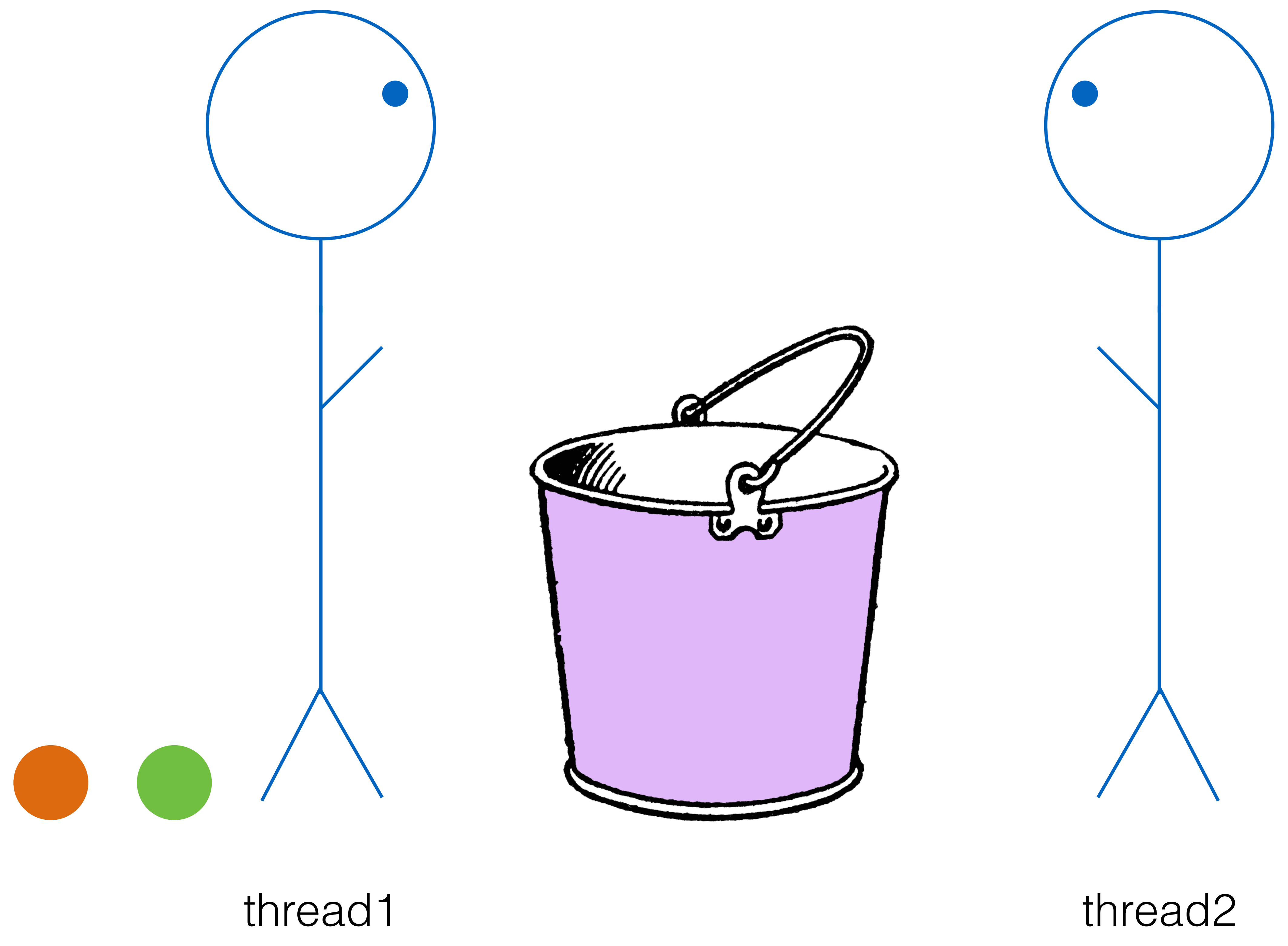
Adds a ball to the bucket, and wakes up any threads that were waiting for one to be added

Semaphores

semaphore.wait() (again)



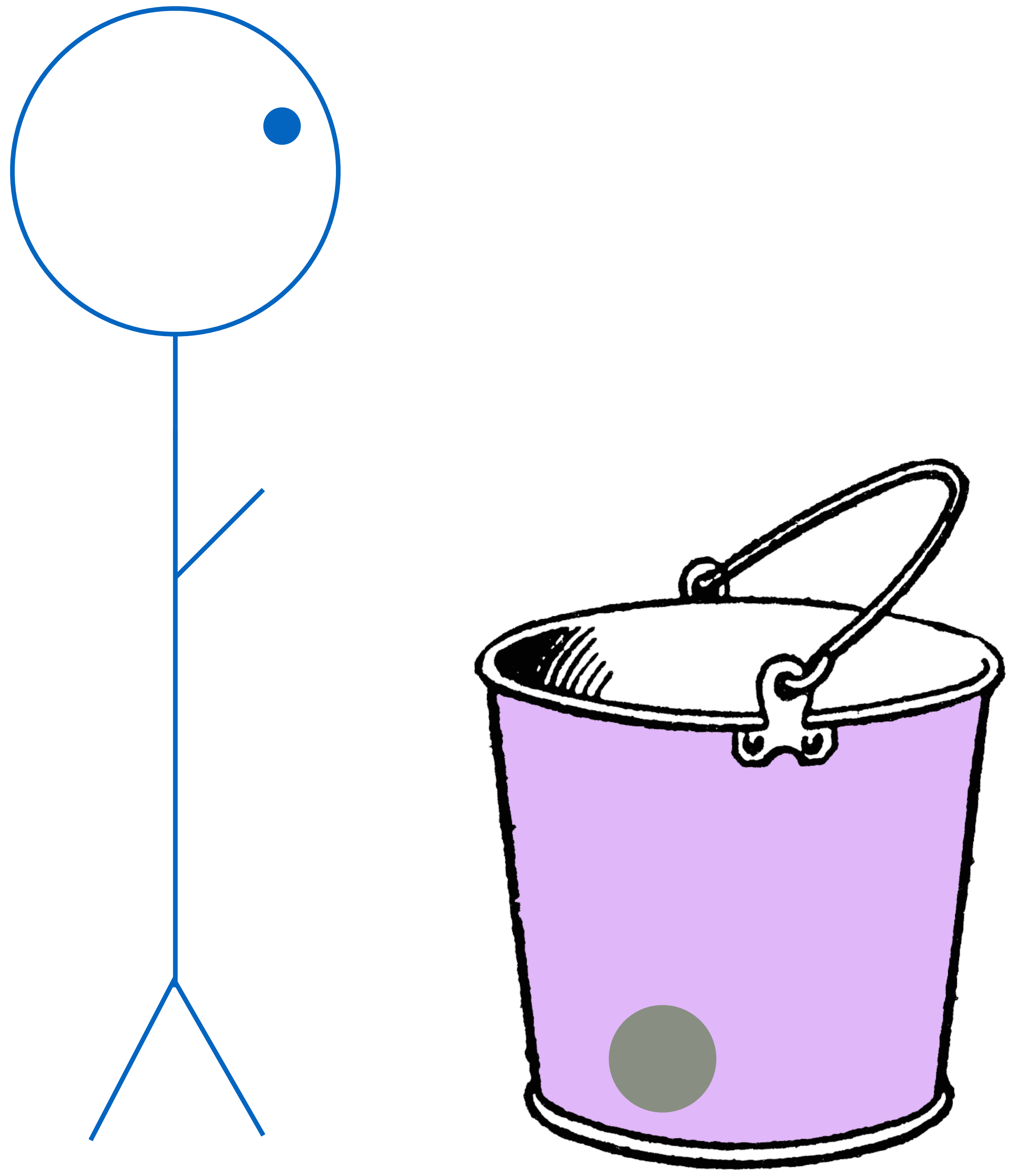
Semaphores



Semaphore methods

- signal():
  - Adds a ball to the bucket
  - **Never blocks**
- wait():
  - If a ball is in the bucket, takes the ball and returns immediately
  - If no ball is in the bucket, waits until one is available, then takes the ball and returns
- There isn’t anything *actually* stored in the bucket. (Under the hood, semaphores are implemented with a simple counter indicating how many “balls” (or whatever) are in the bucket.) But they are very useful for synchronizing between threads

Producer-consumer: transferring data between threads



Mutex: Unlocked

Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads

semaphore.wait()
thread1

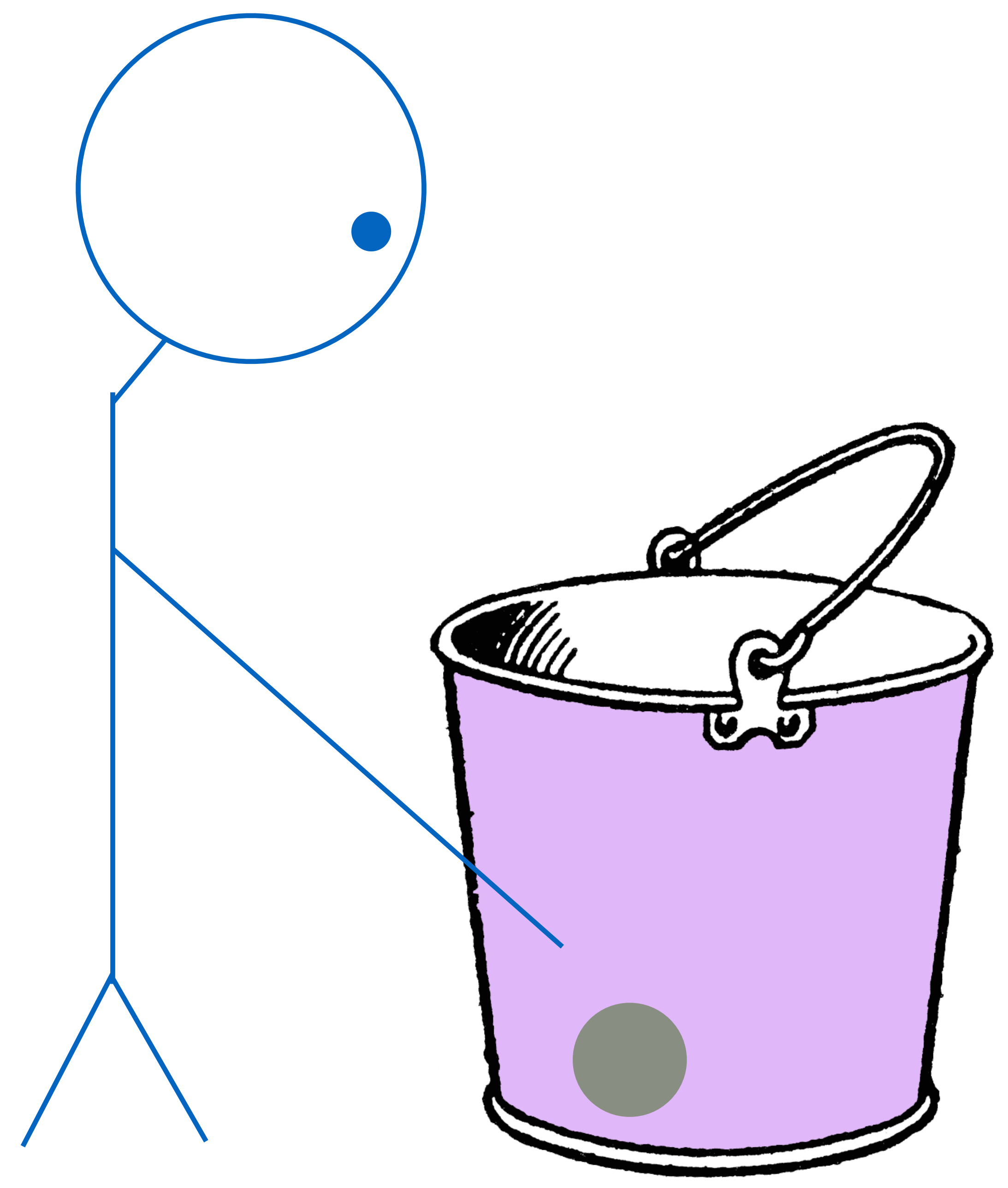
Mutex: Unlocked

Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads

semaphore.wait()



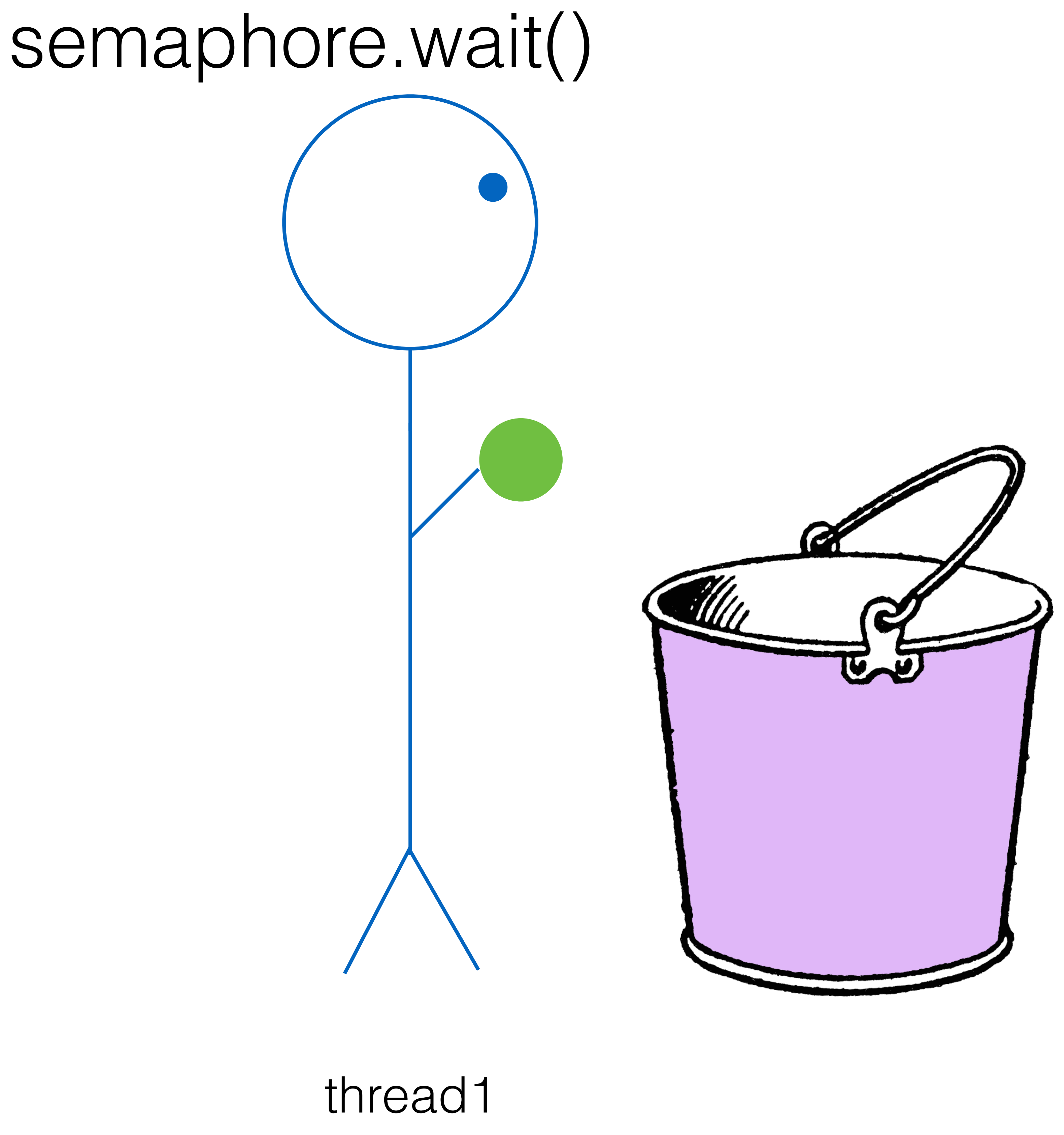
thread1

Mutex: Unlocked

Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads

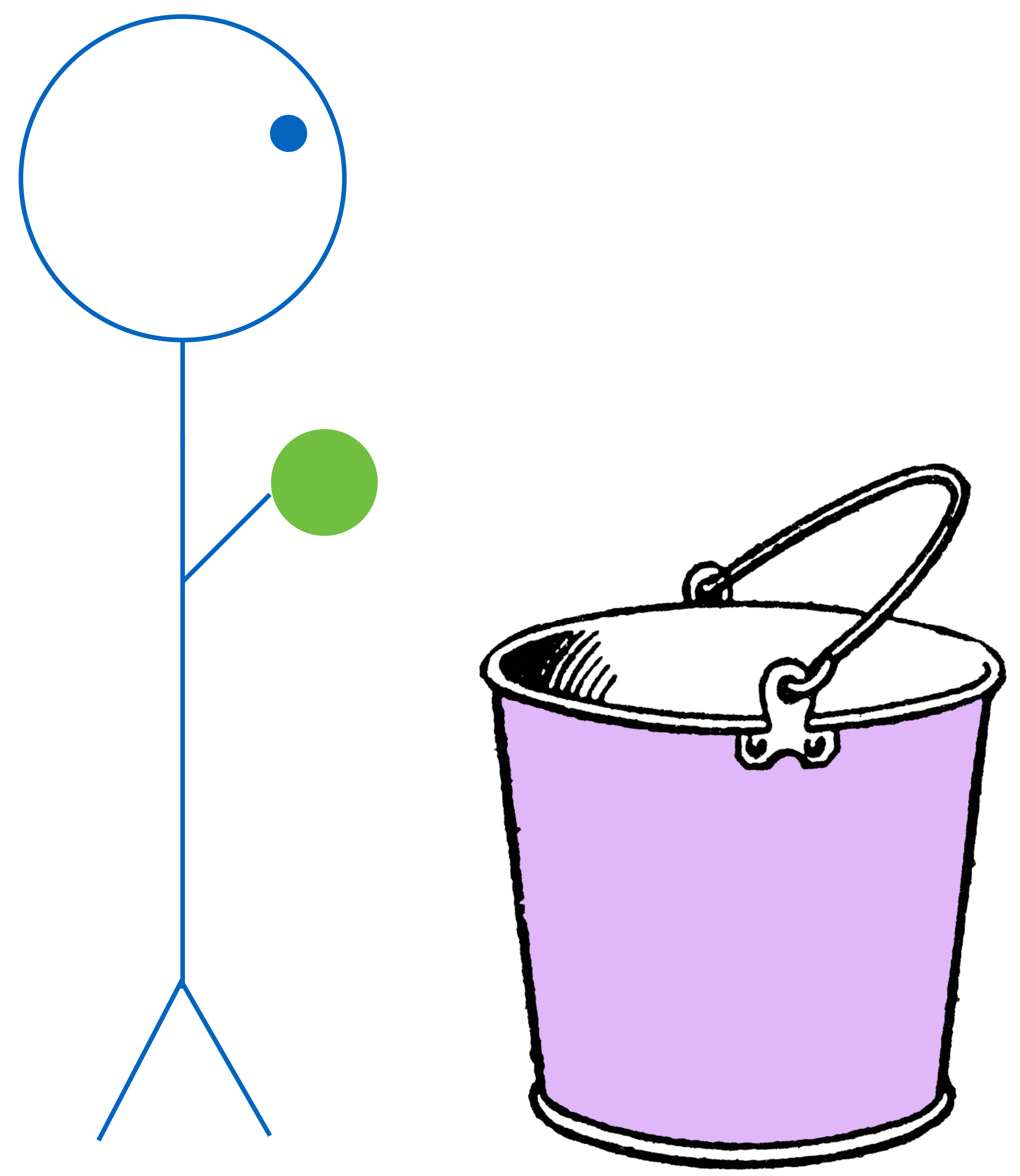


Mutex: Unlocked

Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads

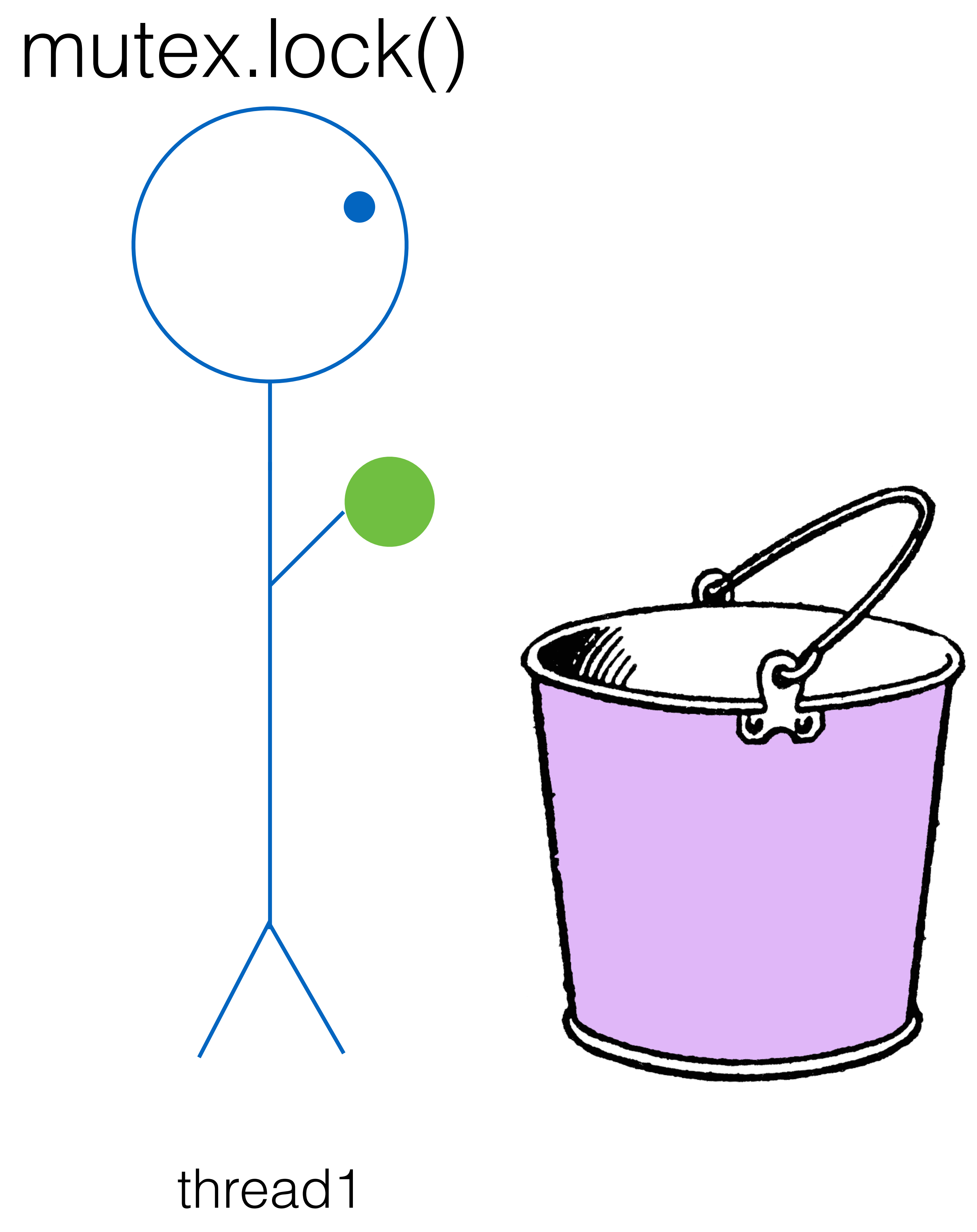


Mutex: Unlocked

Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads



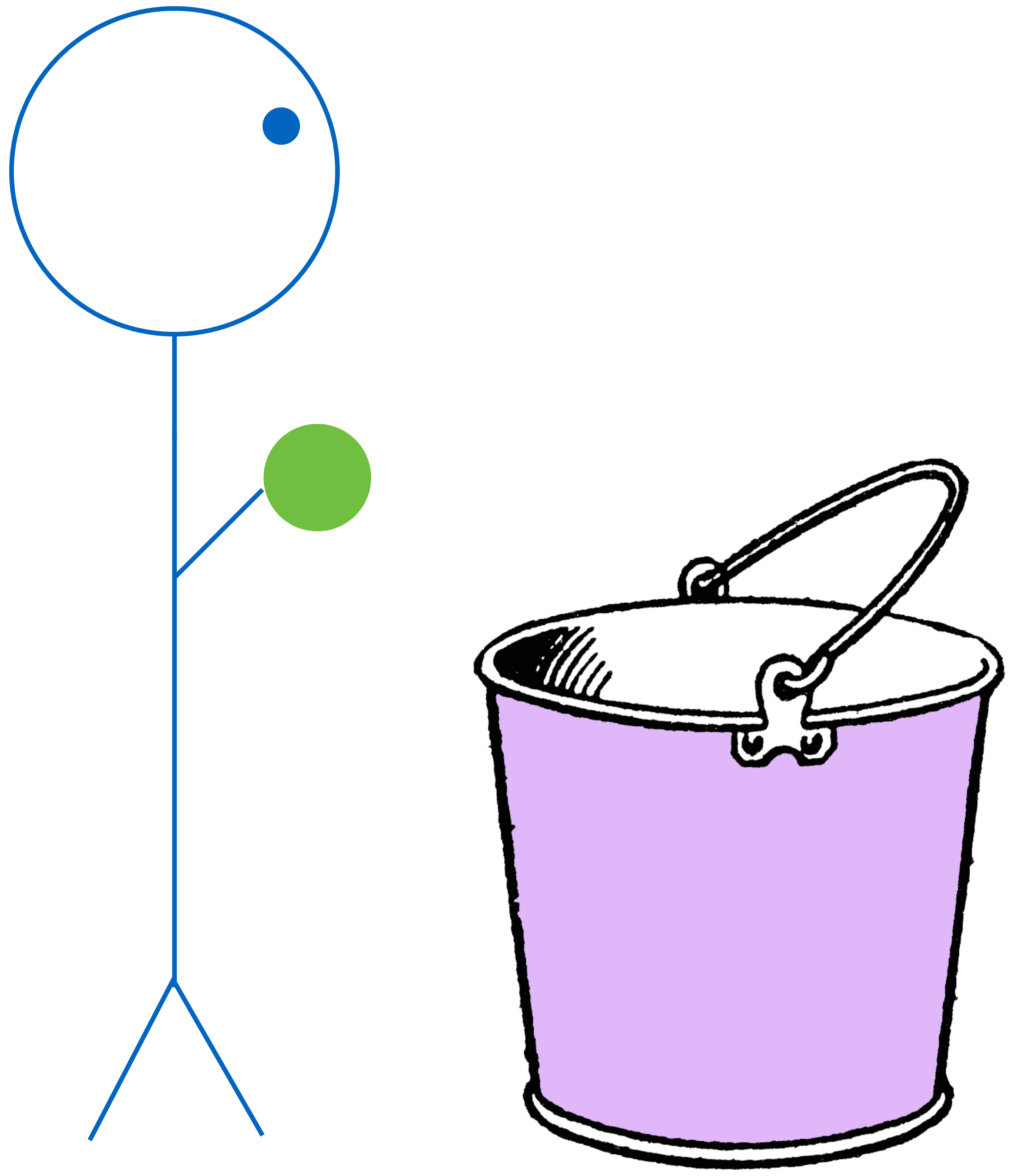
Mutex: Locked

Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads

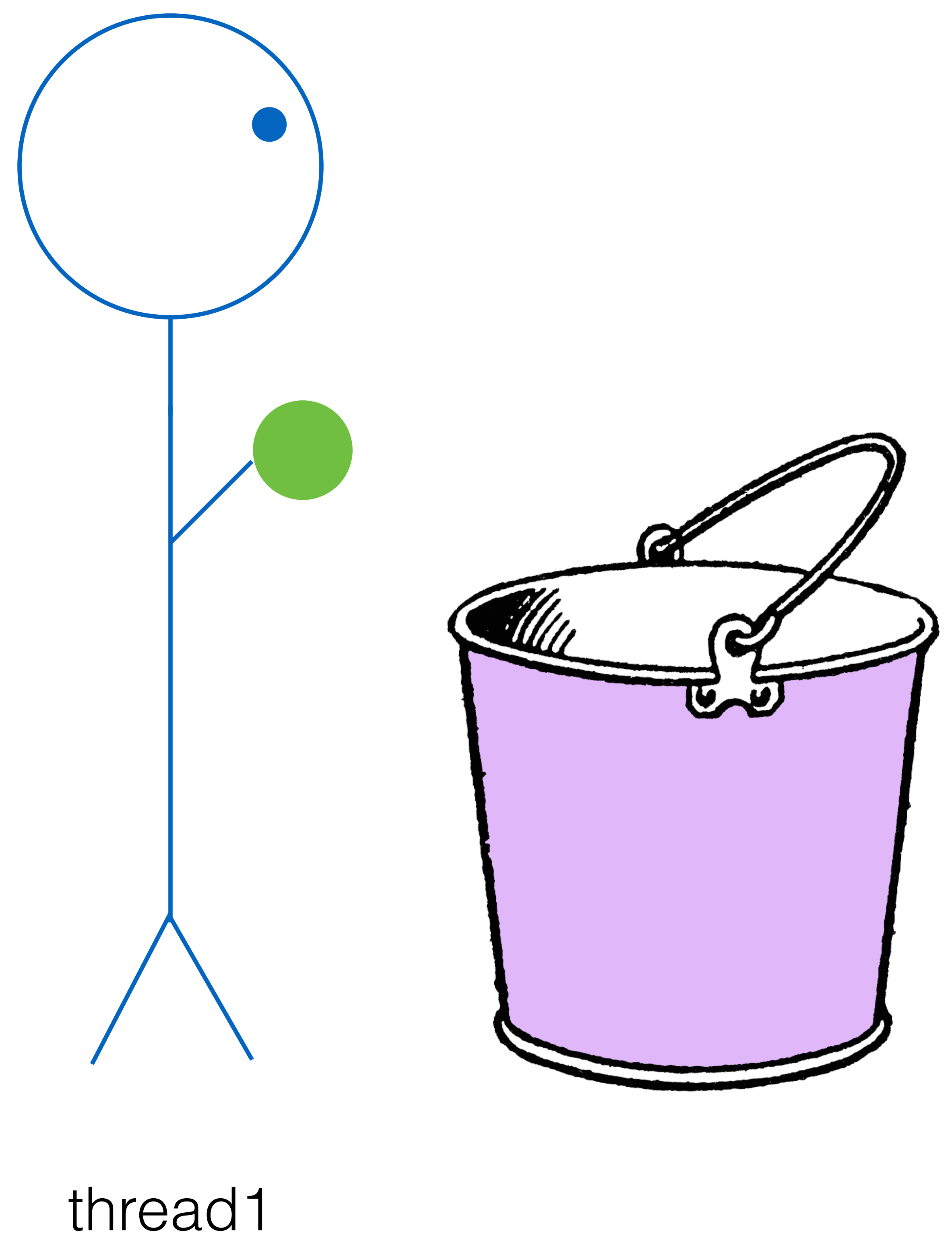
```
SomeStruct {
    ...
}
```



Mutex: Locked

Buffer:

Producer-consumer: transferring data between threads

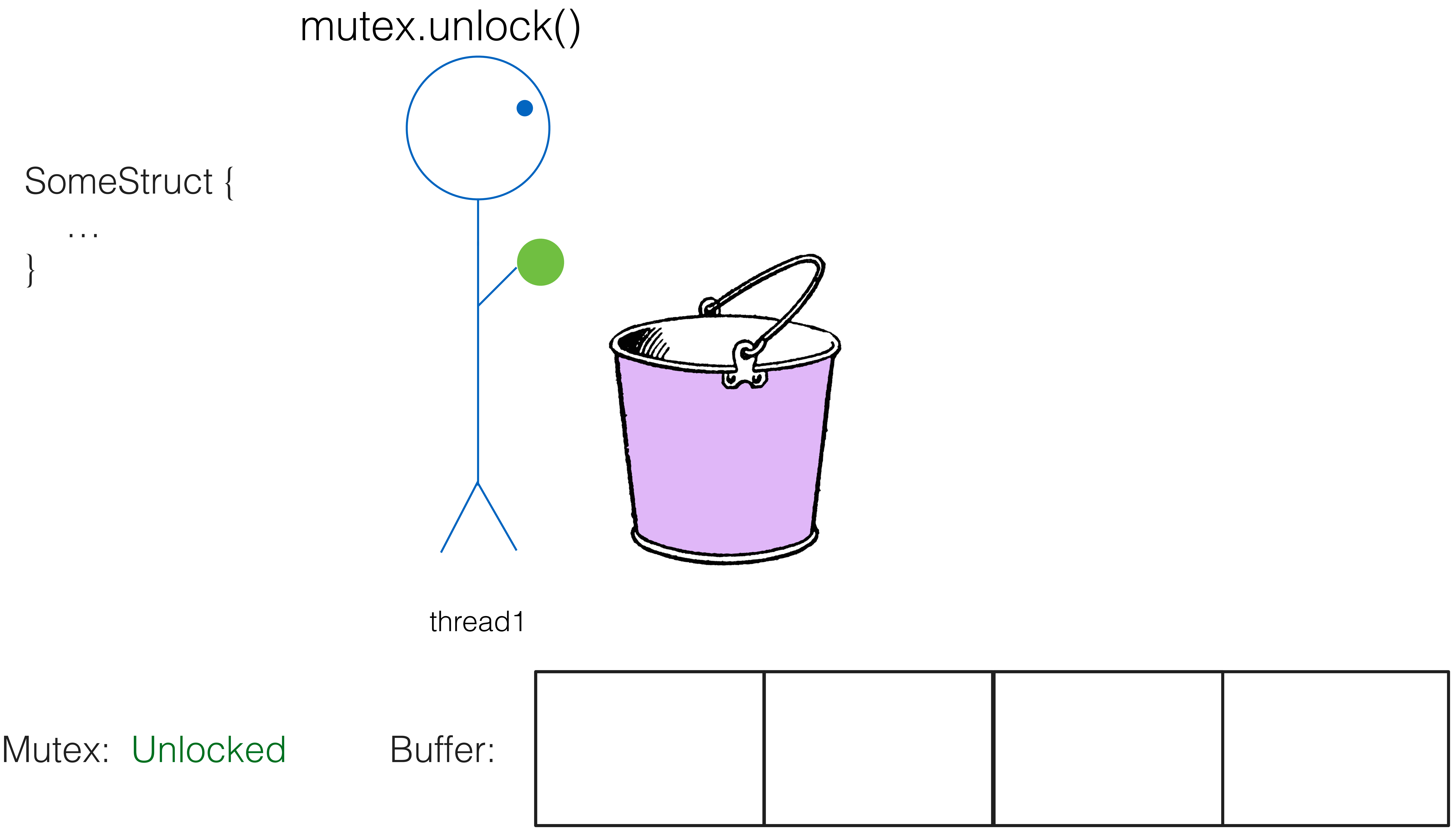


```
SomeStruct {
    ...
}
```

Mutex: Locked

Buffer:

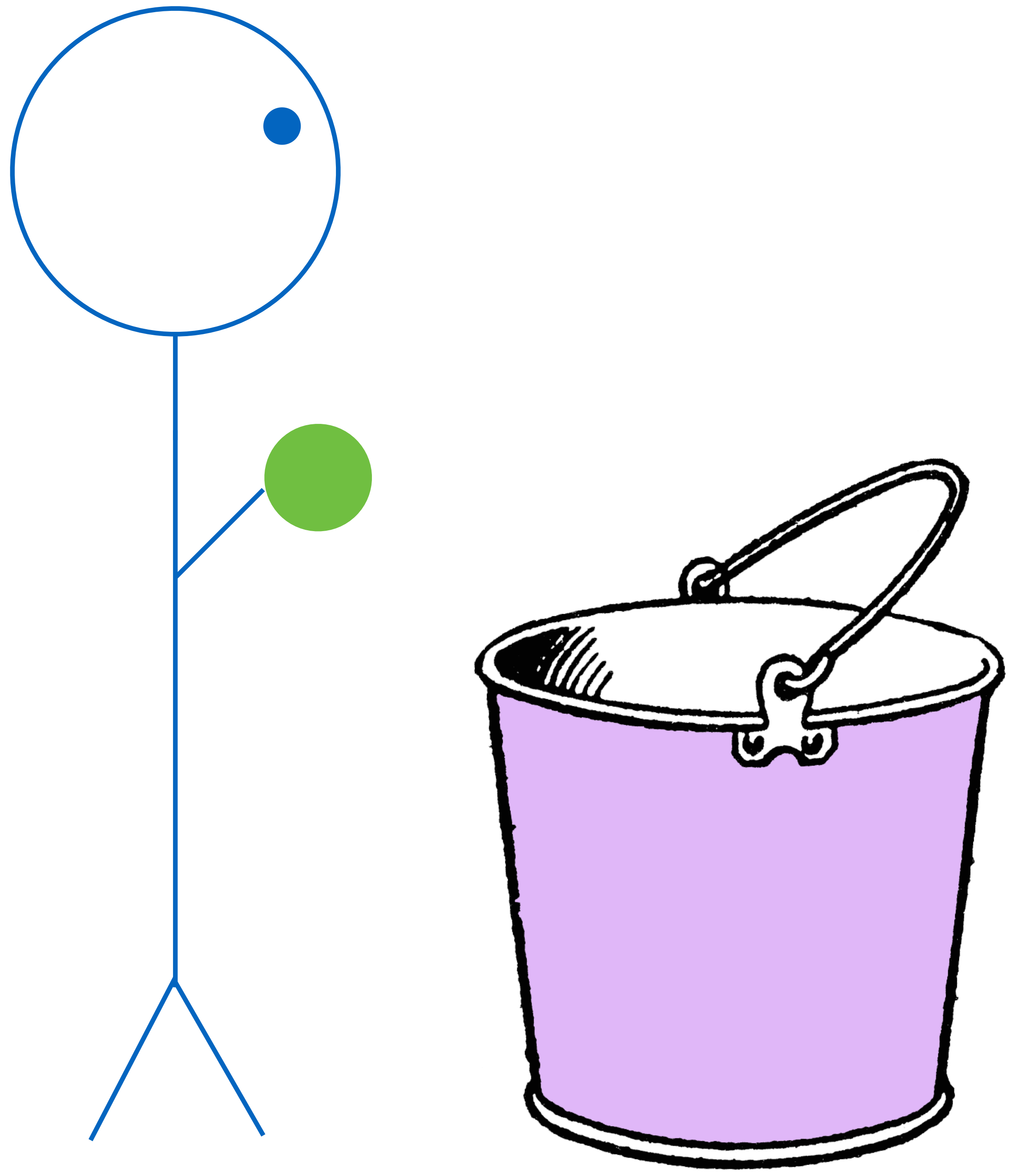
Producer-consumer: transferring data between threads



Producer-consumer: transferring data between threads

semaphore.wait() (again)

```
SomeStruct {
    ...
}
```



thread1

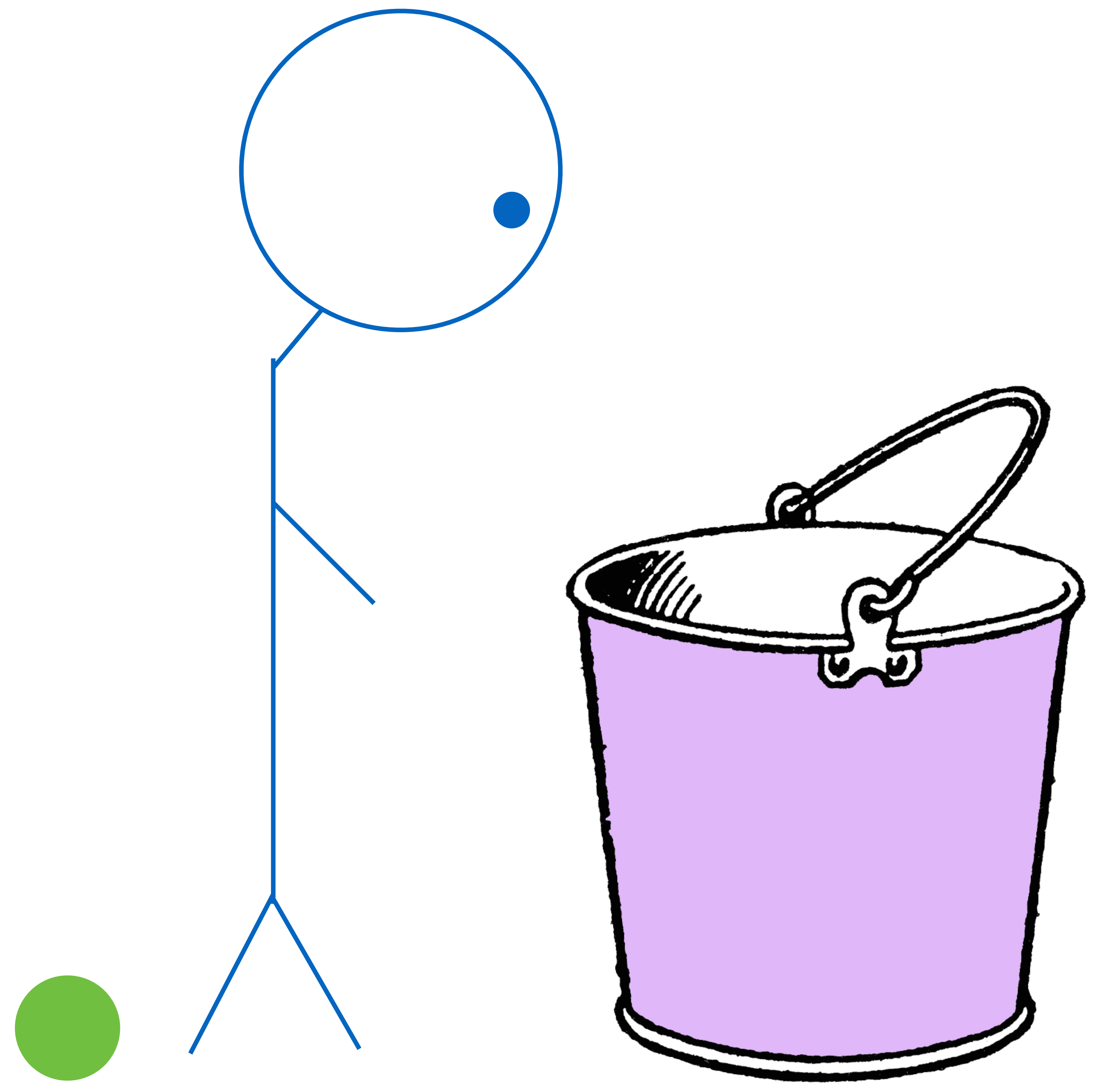
Mutex: Unlocked

Buffer:

Producer-consumer: transferring data between threads

semaphore.wait() (again)

```
SomeStruct {
    ...
}
```



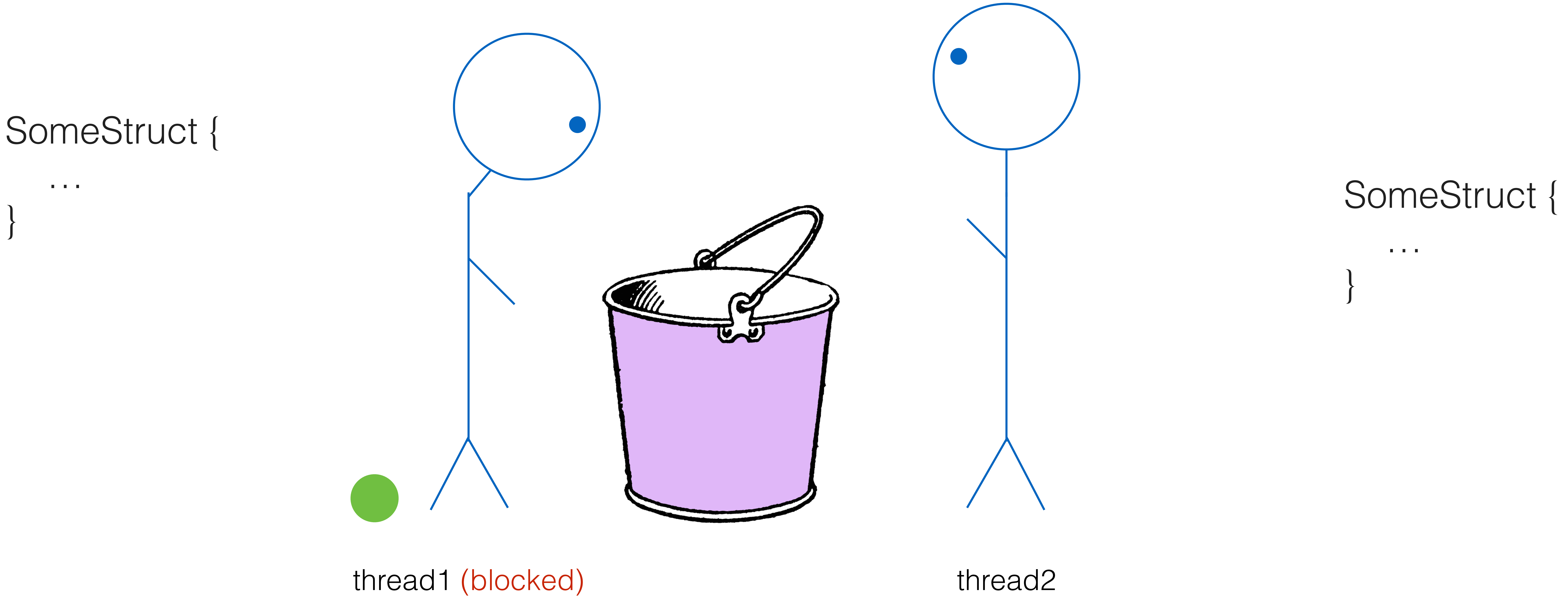
thread1 (blocked)

Mutex: Unlocked

Buffer:

Producer-consumer: transferring data between threads

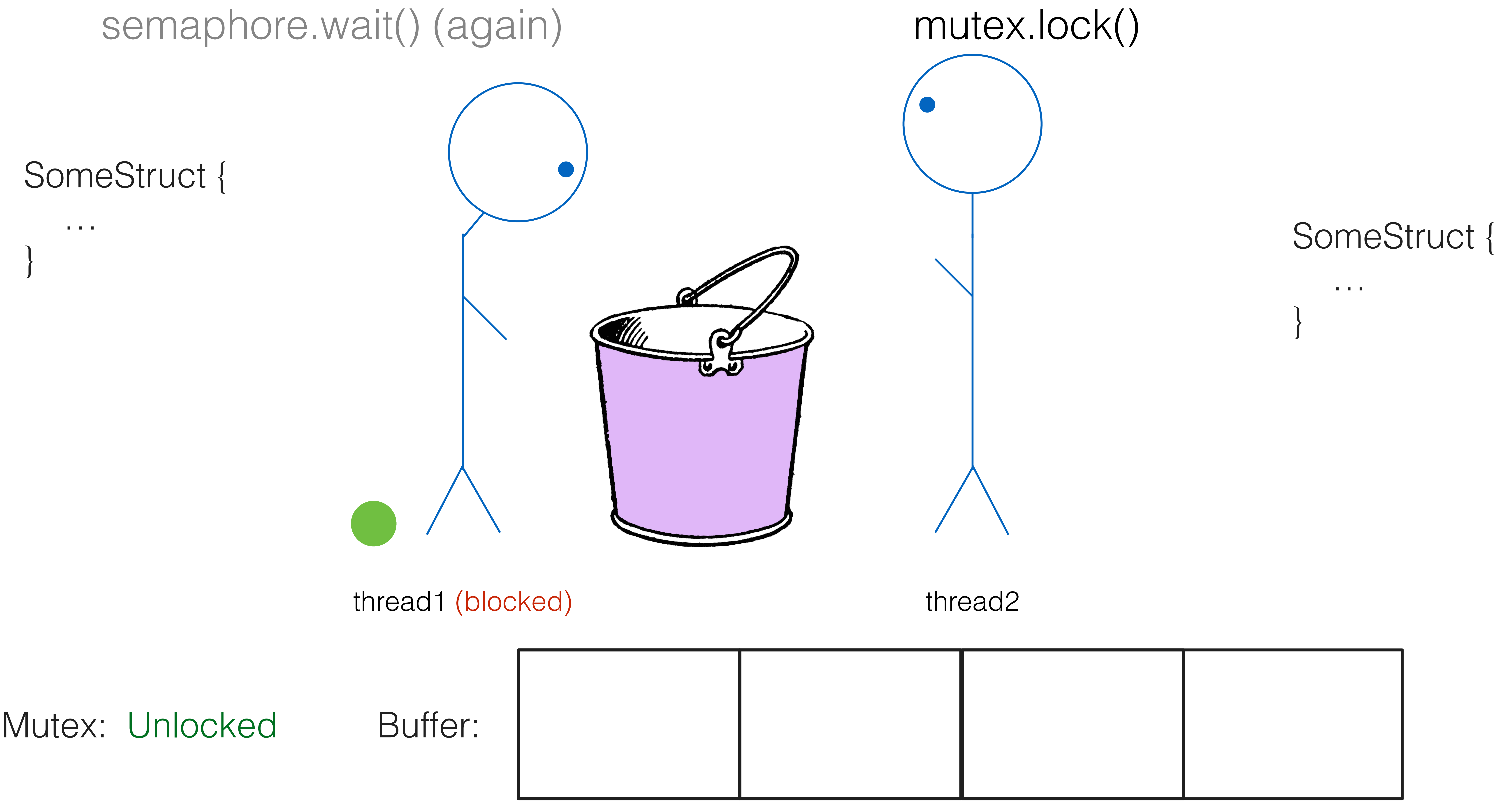
semaphore.wait() (again)



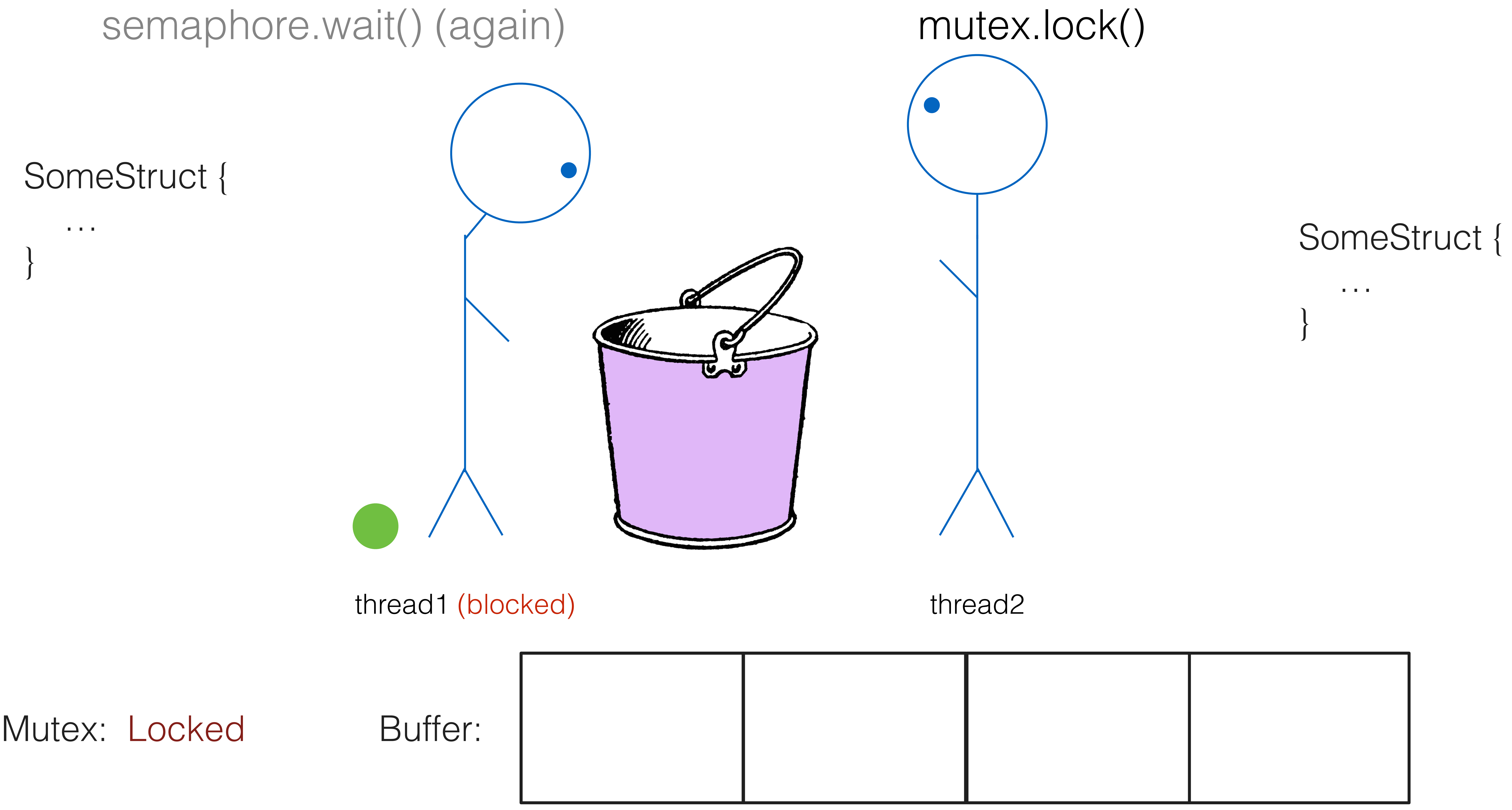
Mutex: Unlocked

Buffer:

Producer-consumer: transferring data between threads



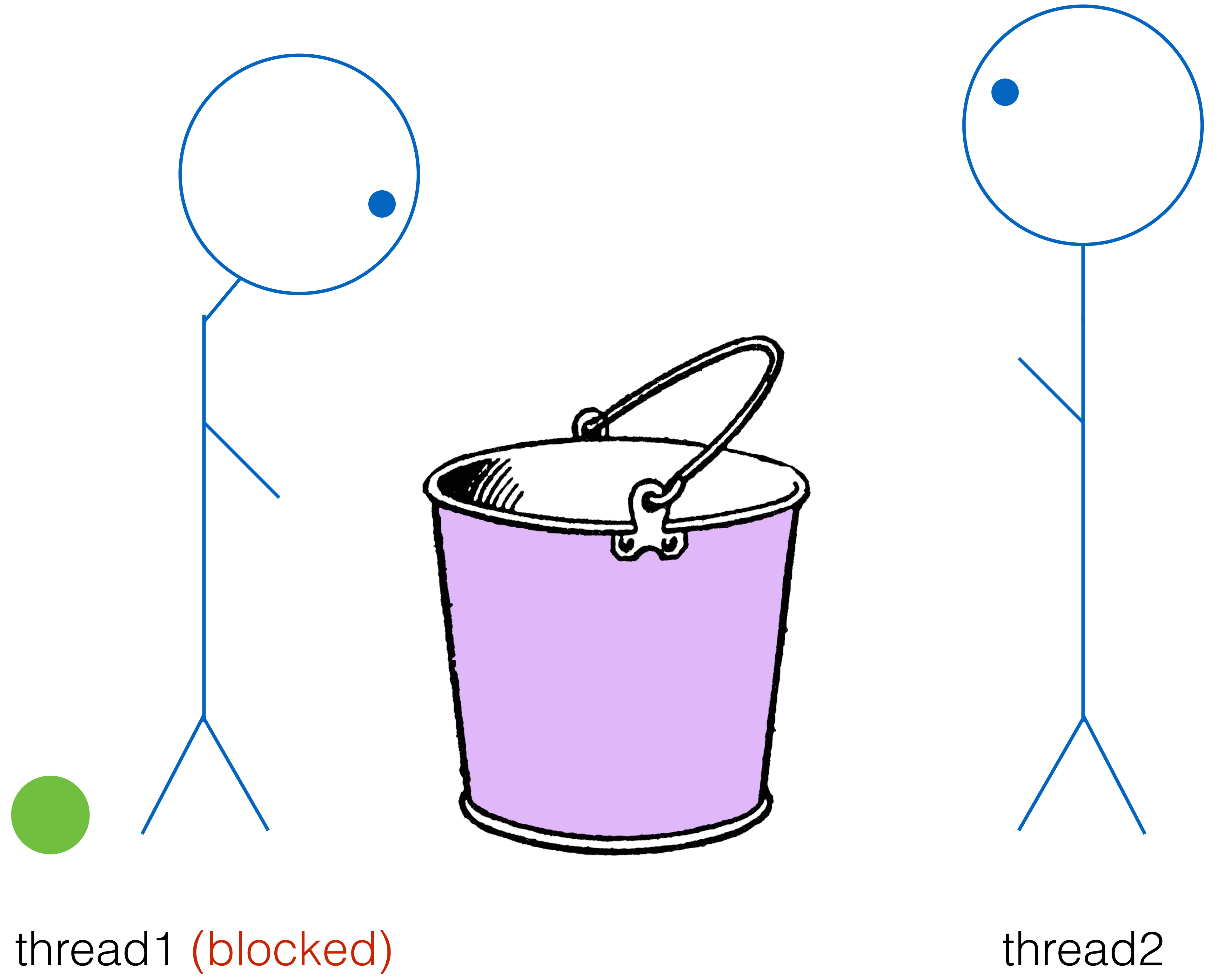
Producer-consumer: transferring data between threads



Producer-consumer: transferring data between threads

semaphore.wait() (again)

```
SomeStruct {
    …
}
```

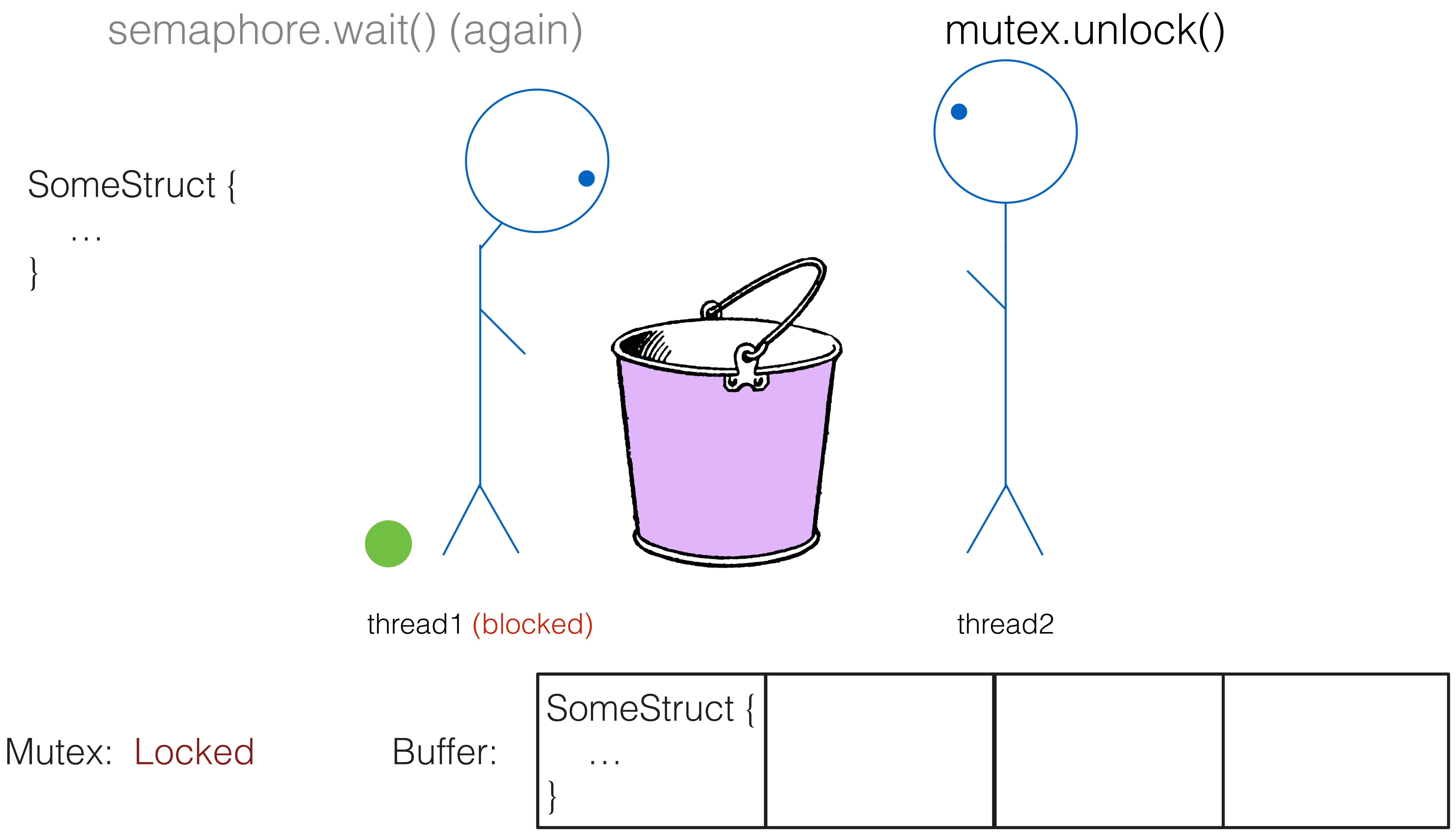


Mutex: Locked

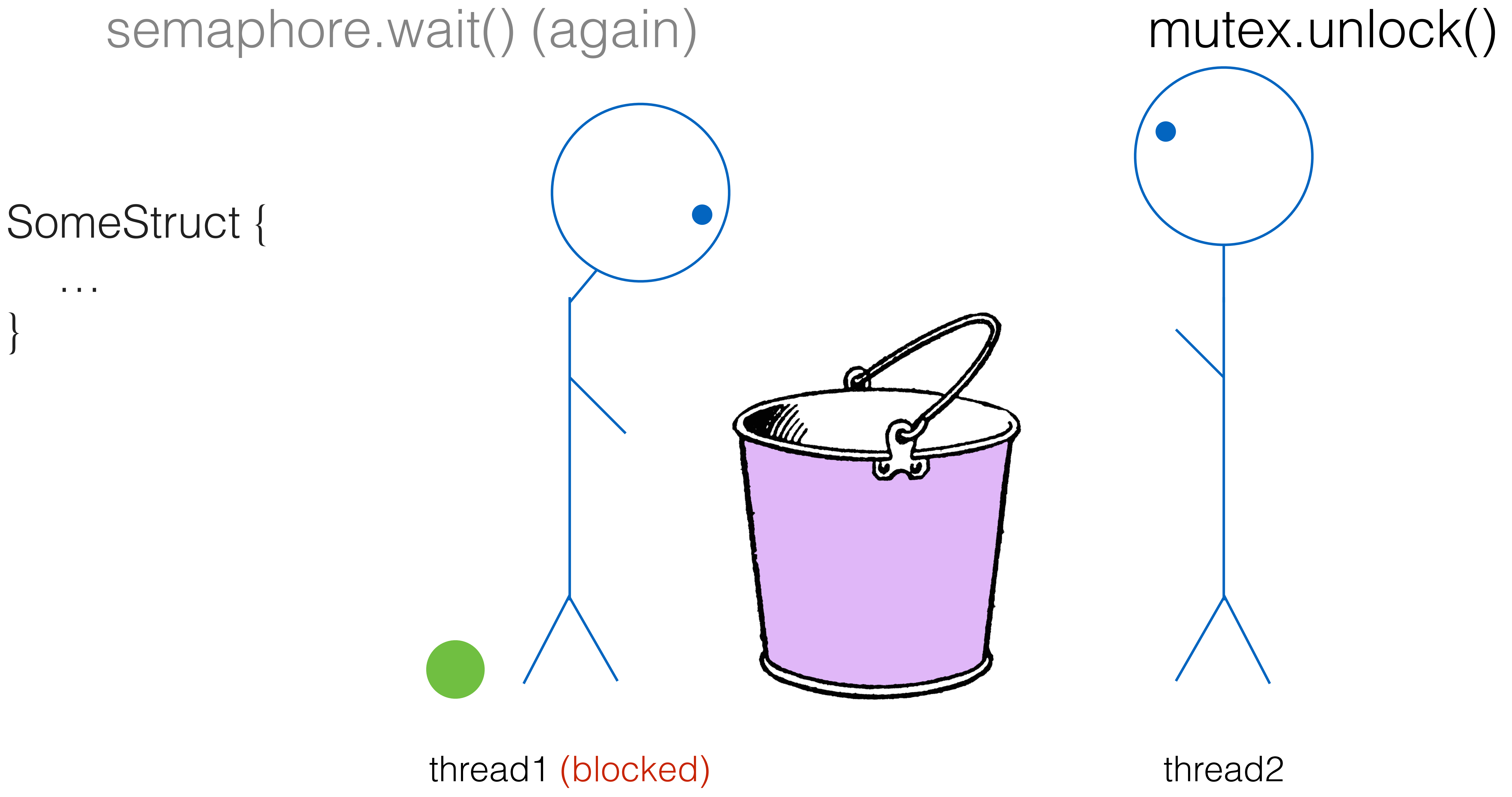
Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads



Producer-consumer: transferring data between threads

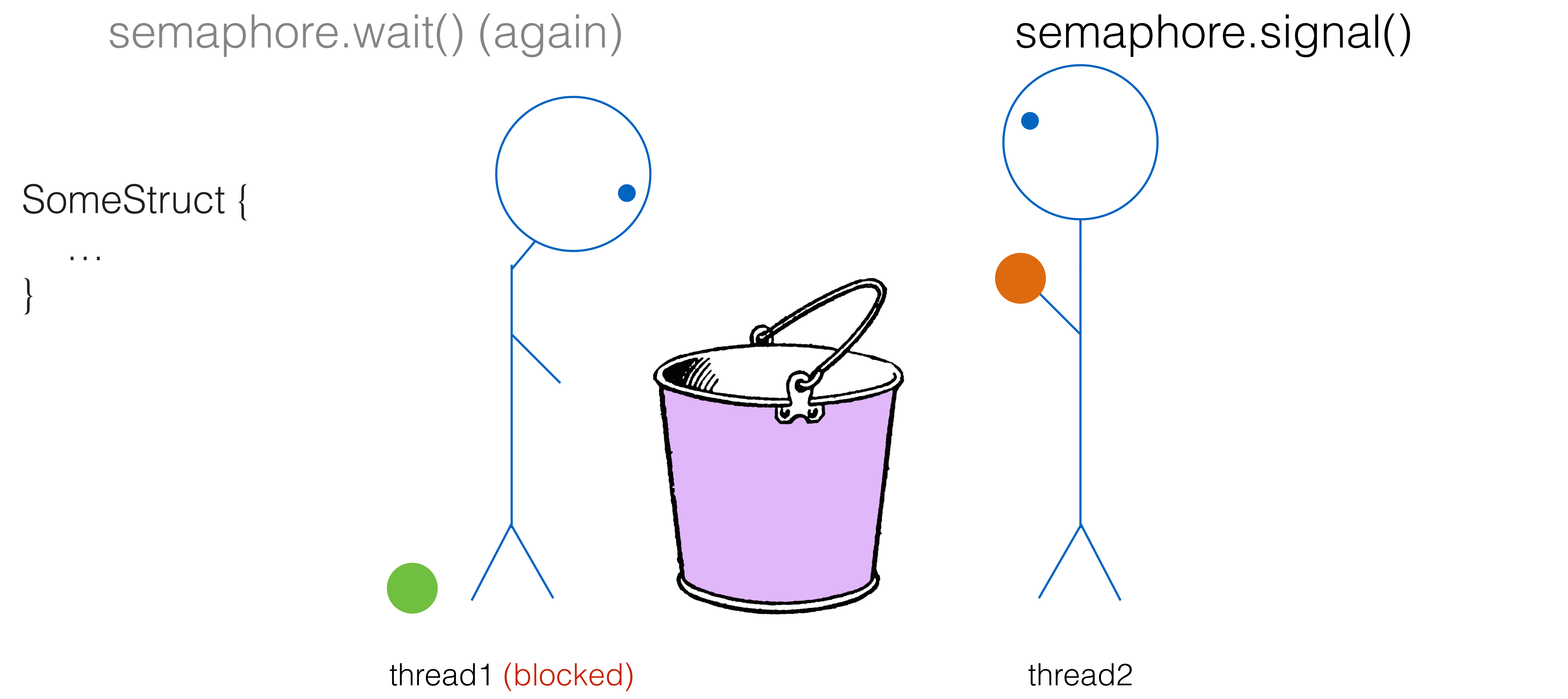


Mutex: Unlocked

Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads

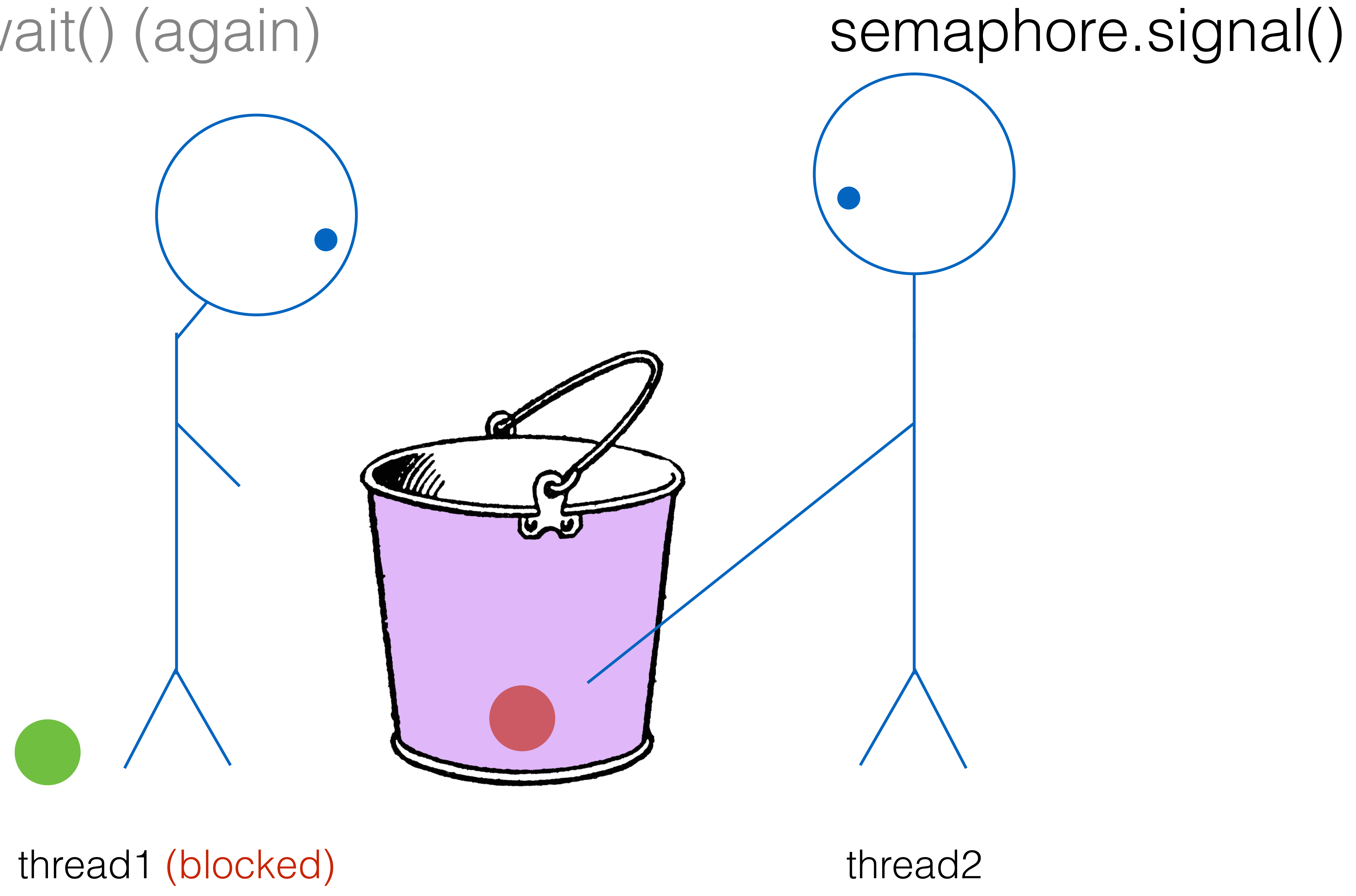


Mutex: Unlocked

Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads



```
SomeStruct {
    …
}
```

Mutex: Unlocked

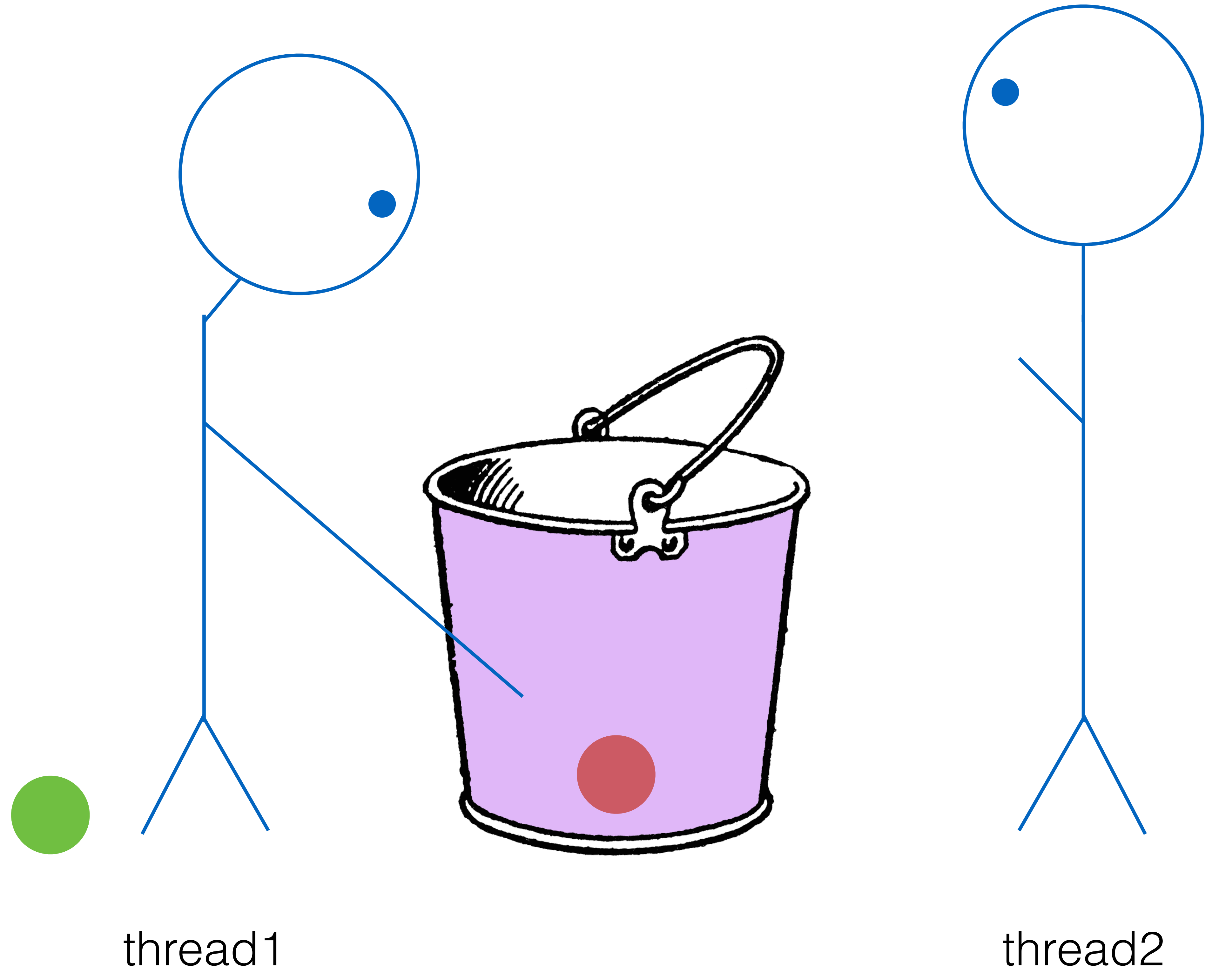
Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads

semaphore.wait() (again)

```
SomeStruct {
    …
}
```



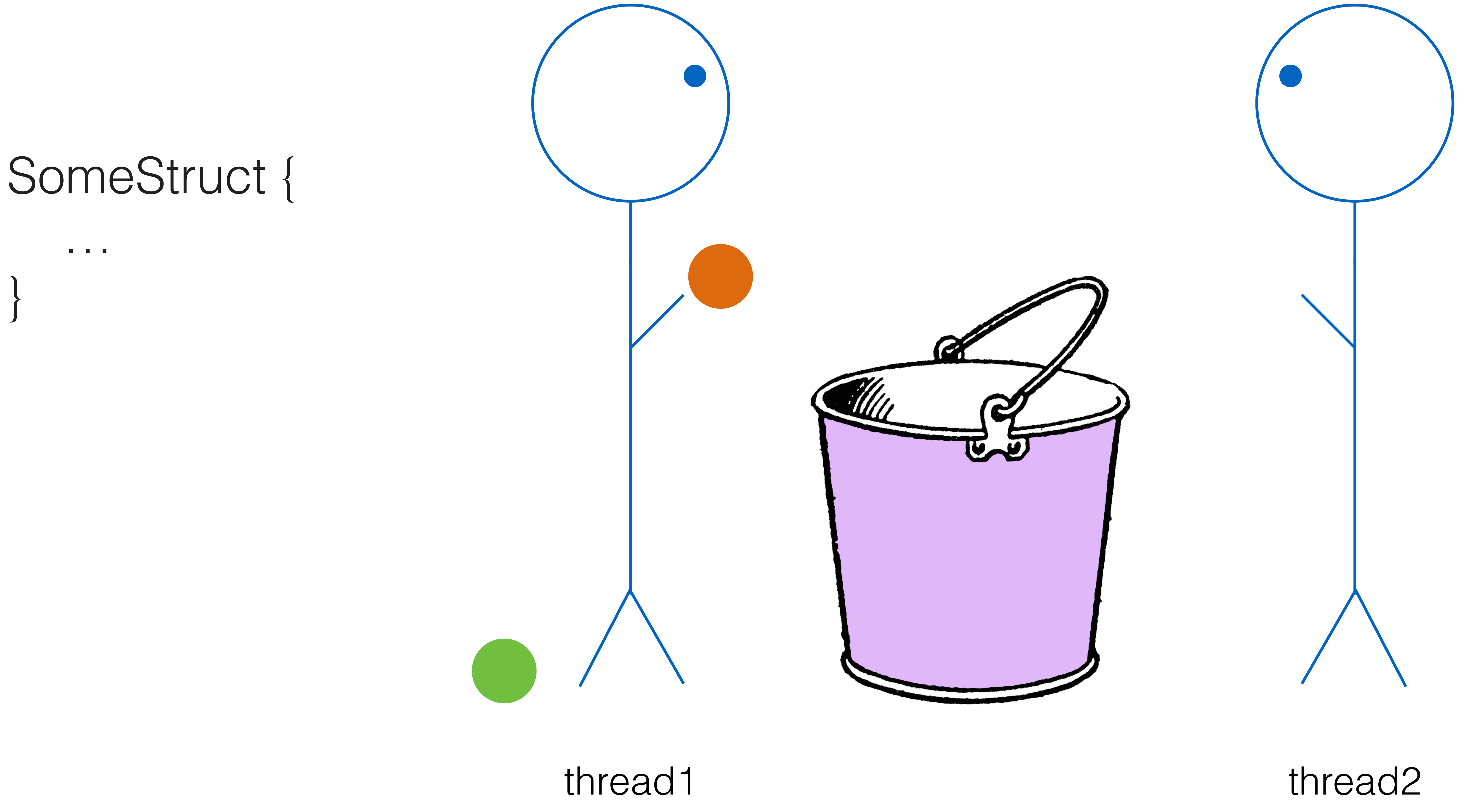
Mutex: Unlocked

Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads

semaphore.wait() (again)

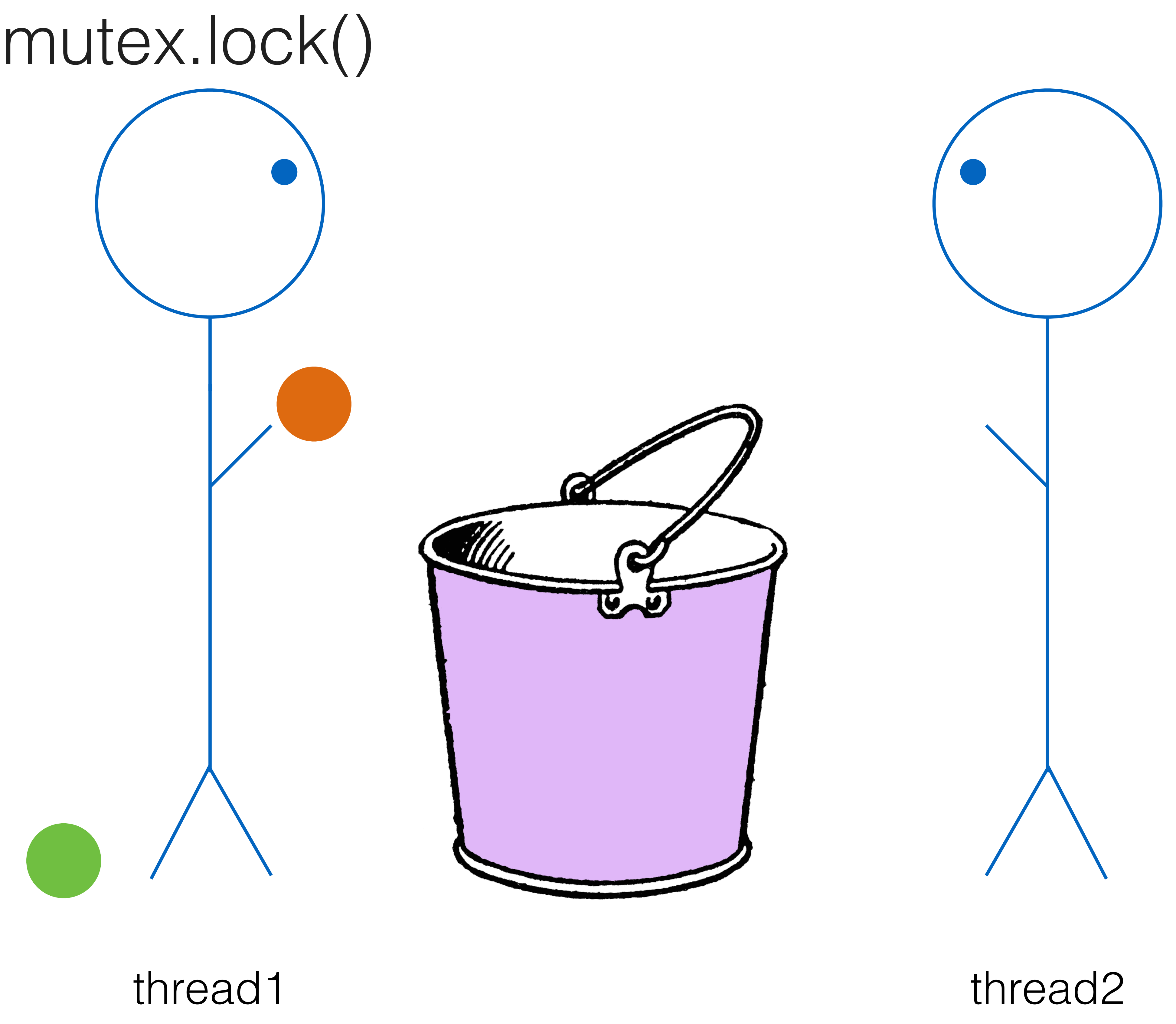


Mutex: Unlocked

Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads

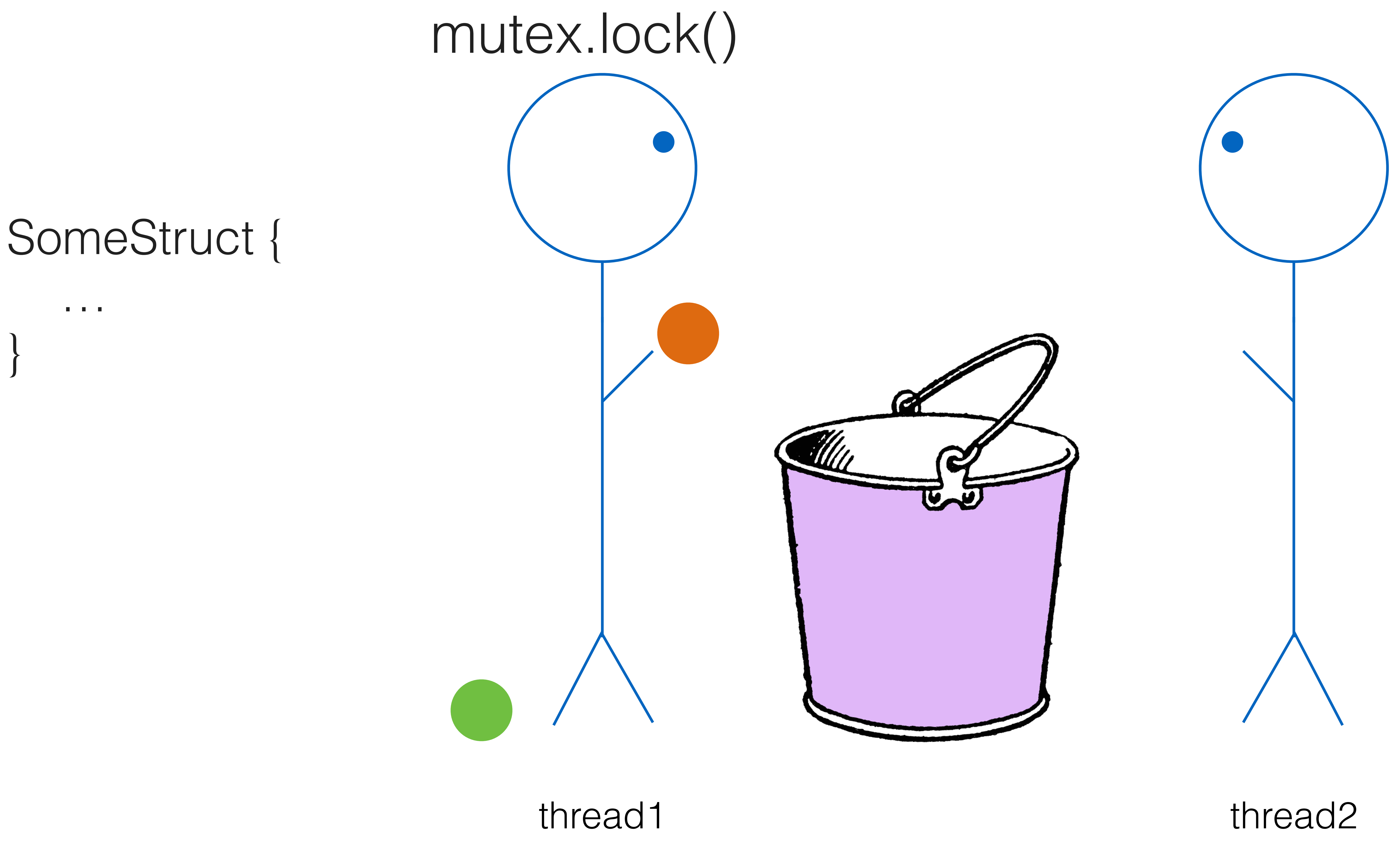


SomeStruct {
    …
}

Mutex: Unlocked   Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads



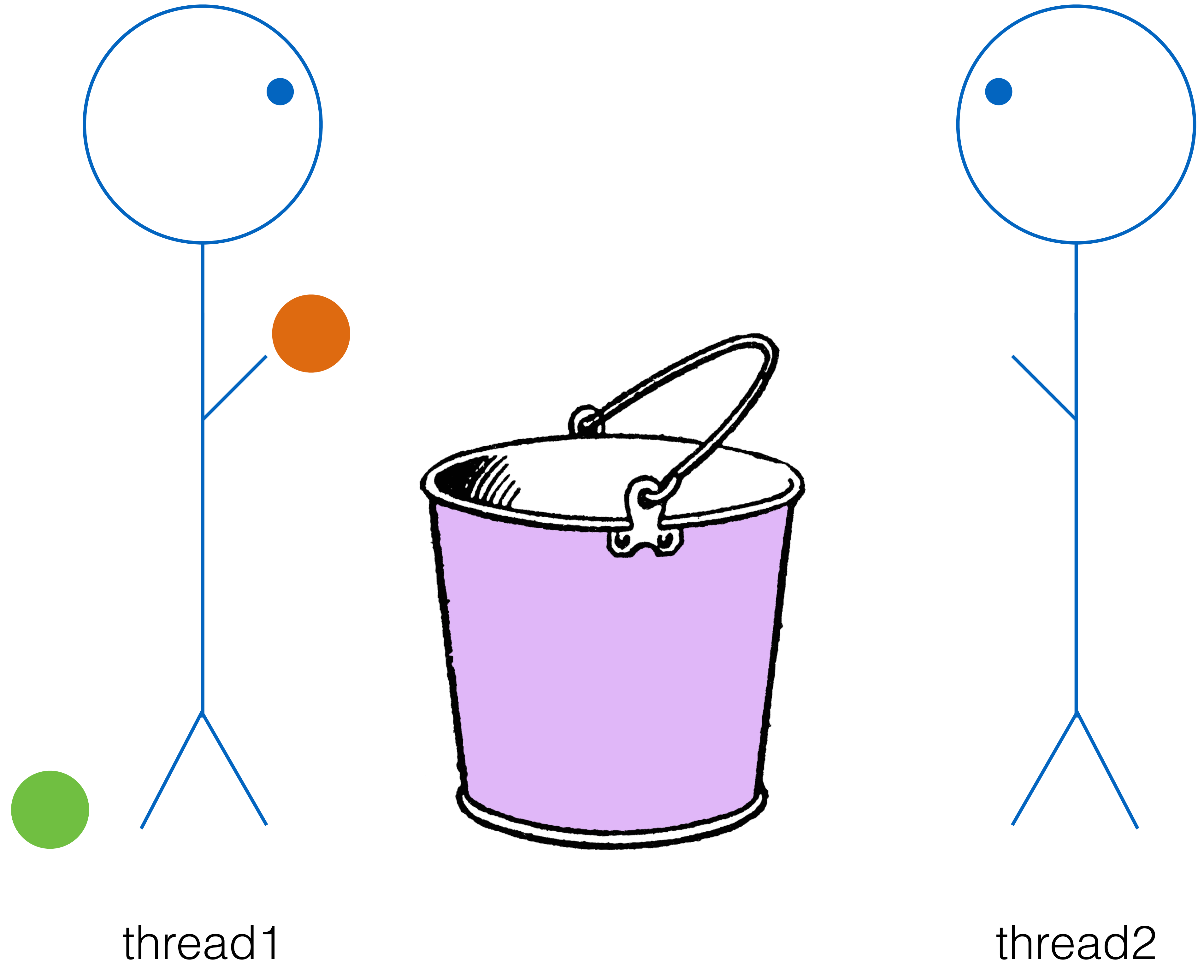
Mutex: Locked

Buffer:

SomeStruct { … }			

Producer-consumer: transferring data between threads

```
SomeStruct {
    ...
}

SomeStruct {
    ...
}
```



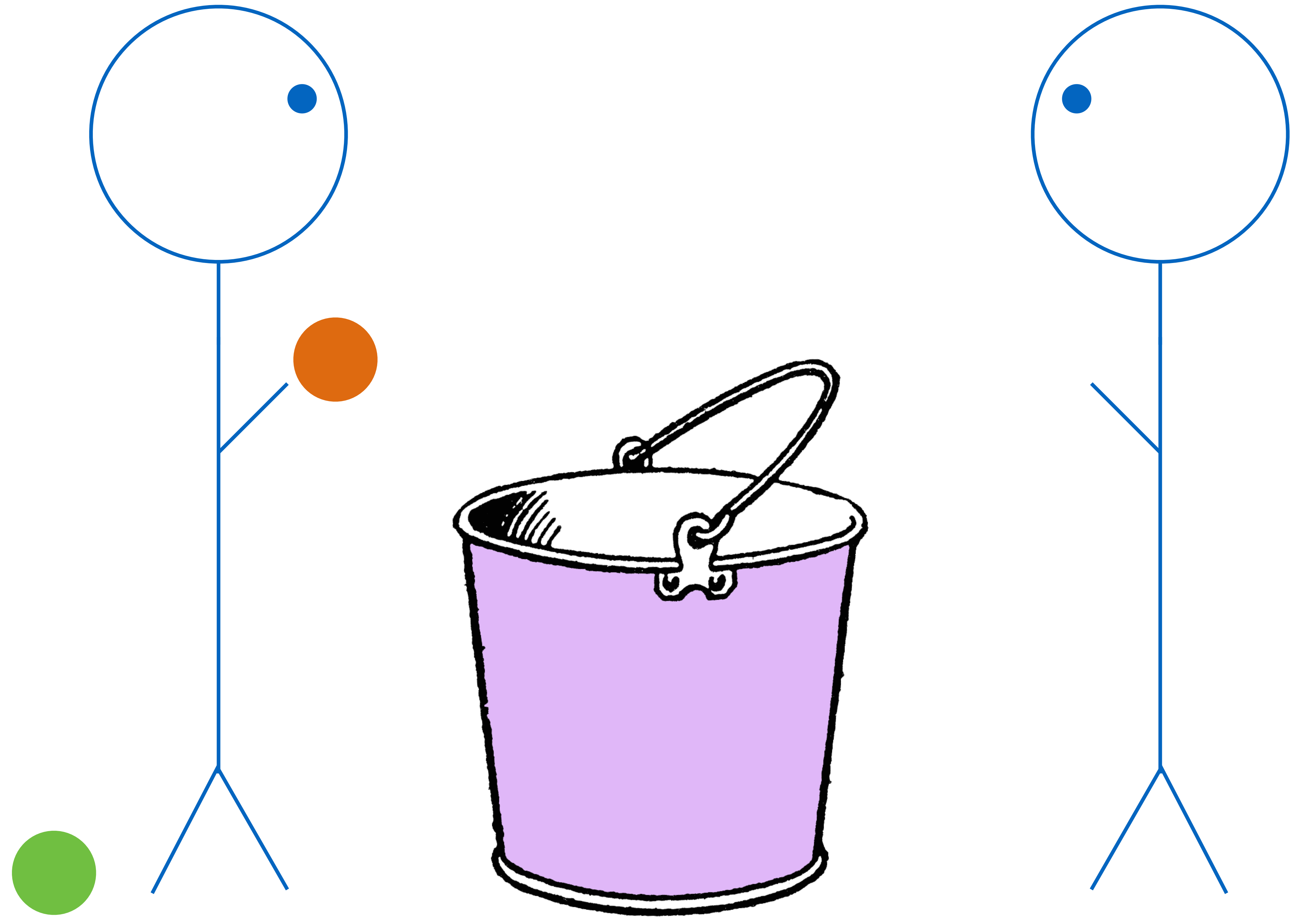
Mutex: Locked

Buffer:

Producer-consumer: transferring data between threads

mutex.unlock()

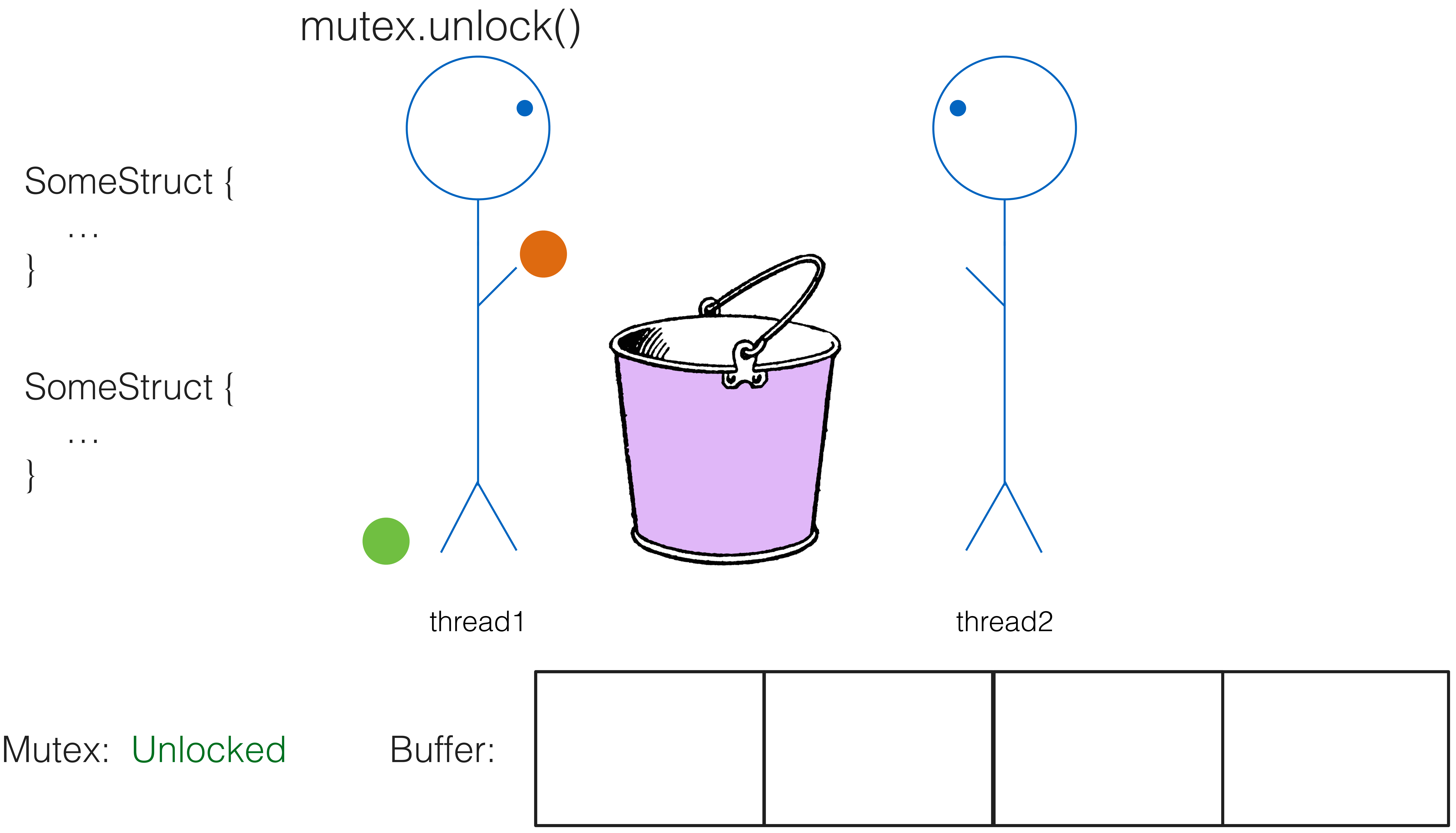
```
SomeStruct {
    …
}

SomeStruct {
    …
}
```



Mutex: Locked

Buffer:

Producer-consumer: transferring data between threads


mutex.unlock()
SomeStruct {
...
}
SomeStruct {
...
}
thread1
thread2
Mutex: Unlocked
Buffer: